A PRIMERA REVISTA DE PROGRAMACIÓN EN CASTELLANO

SOLO PROGRAMADORES

O V, NÚMERO 47

975 Ptas.

A LIBRERÍA DE JUEGOS

LINUX

SVGALIB

REDES LOCALES
El arranque de Windows NT

C/C++
Programación básica en Windows

JAVA
Programación de servlets (y III)

PROGRAMACIÓN DE SOCKETS
Integración de CGI en el servidor web (III)

PROGRAMACIÓN CIENTÍFICA
Representación gráfica de funciones 3D

CONTENIDO DEL CD-ROM

- TCL/TK v8.0
- Visual TCL v1.11
- Magic v8.01
- NDS for NT v1.0
- GNU Network Object Model Environment
- NetShow Theater Server

TOWER

8 413042 303299

SÓLO PROGRAMADORES

Número 47
SÓLO PROGRAMADORES
es una publicación de
TOWER COMMUNICATIONS

Director Editor
Antonio M. Ferrer Abelló
aferrer@towercom.es
Coordinador Técnico
Eduardo de Riquer
eriquer@towercom.es
Redactora Jefe
Cristina Peña
crisp@towercom.es

Colaboradores
Enrique Díaz, Ernesto Schmitz, Chema Alvarez, Constantino Sánchez Ballesteros, Enrique de la Lastra, Jordi Agost, Juan Carlos Sastre, Enrique de Alarcón, Jorge Delgado, Manel Guerrero.

Maquetación
Susana Llano
Tratamiento de Imagen
María Arce Giménez

Publicidad
Erika de la Riva (Madrid)
Tel.: (91) 661 42 11
Pepín Gallardo (Barcelona)
Tel.: (93) 213 42 29 - 970 47 44 35

Suscripciones
Begoña Alvarez
Tel. (91) 661 42 11 Fax: (91) 661 43 86
suscrip@towercom.es

Laboratorio
Óscar Rodríguez (jefe)
oscarri@towercom.es
Javier Amado
jamado@towercom.es
Servicio Técnico
Óscar Casado
ocasado@towercom.es

Filmación
Megatipo
Impresión
Gráficas Reunidas
Distribución
SGEL
Distribución en Argentina / Chile / Colombia / México / Venezuela
Capital: Huesca y Sanabria
Interior: D.G.P.

TOWER COMMUNICATIONS
Director General
Antonio M. Ferrer Abelló
Director Financiero
Francisco García Díaz de Liaño
Director de Producción
Carlos Peropadre
Directora Comercial
Carmina Ferrer
carmina@towercom.es

Redacción, Publicidad y Administración
C/ Aragoneses, 7
28108 Pol. Ind. Alcobendas (MADRID)
Telf.: (91) 661 42 11 / Fax: (91) 661 43 86

La revista Sólo Programadores no tiene por qué estar de acuerdo con las opiniones escritas por sus colaboradores en los artículos firmados. El editor prohibe expresamente la reproducción total o parcial de los contenidos de la revista sin su autorización escrita.

Depósito legal: M-26827-1994
ISSN: 1134-4792
PRINTED IN SPAIN
COPYRIGHT 30-09-98

EDITORIAL

A Merced de la tecnología

Cualquier programador con algunos años de experiencia a sus espaldas, más de una vez habrá reflexionado: "cada año que pasa tengo una preparación más completa, pero en realidad sé bastante menos". Aunque parezca paradójico, esta situación tan familiar para muchos se muestra como un aspecto intrínseco del mundo de la tecnología y por tanto de la informática. Antes teniendo conocimientos de COBOL, bases de datos y poco más, te consideraban el rey de los ordenadores. Por aquel entonces, nada se nos resistía. Supongo que esta situación siempre ha existido, pero debido al avance veloz e imparable de la tecnología, el tiempo de reacción y adaptación resulta casi exiguo. Lo que sí está claro es que el abanico de lenguajes, herramientas, sistemas operativos, interfaces, procesadores, tipologías de red y derivados de Internet, contiene una amplitud de proporciones difíciles de asimilar con éxito.

Las nuevas tecnología poseen la virtud de solucionar algunos de nuestros problemas, de facilitar ciertas tareas tediosas y a la vez, de constituir un nuevo reto para todos los programadores (y usuarios en general). El gran público la recibe con cuentagotas, eso sí, con un ritmo firme y continuo, al que el software sólo puede observar desde el rebufo. Pero, ¿dónde se encuentra el verdadero aprovechamiento de la tecnología y dónde se esconde el oportunismo del marketing? ¿Cuándo se aprovechan realmente los avances tecnológicos orientados a la programación? Supongo que la barrera que separa un concepto de otro resulta demasiado difusa como para aventurarse, pero existen ejemplos claros donde la línea separadora sí debería apreciarse con claridad.

El lanzamiento del procesador Merced de Intel con frecuencias de reloj que cuesta asimilar, un ancho de banda de 64 bits y la nueva tecnología de arquitectura EPIC (*Explicit Parallel Instruction Computing*), se presenta como uno de esos ejemplos donde las actualizaciones merecerán la pena. A grandes rasgos, una de las ventajas directas de EPIC reside en la capacidad del procesador para ejecutar varias microinstrucciones, basada en métodos de predicción y especulación que permiten anticiparse a las necesidades del programa. El procesador intentará adelantarse a dichas necesidades trabajando en varios frentes mientras espera datos de otro componente. Así se obtiene un mayor aprovechamiento de la CPU, así como una mayor eficiencia en la ejecución de aplicaciones.

Tenéis a vuestra disposición la dirección de correo electrónico de la revista: solop@towercom.es, donde podéis enviar vuestras sugerencias sobre los contenidos, los temas que queráis que incluyamos y los productos que os gustaría que distribuyéramos con el CD-ROM, que todos los meses regalamos. Sois nuestros mejores colaboradores.

SÓLO PROGRAMADORES 47

6 Noticias,
LAS ÚLTIMAS NOVEDADES EN LA TECNOLOGÍA INFORMÁTICA

Os vamos a contar las últimas novedades en este sector; os ponemos al día de las nuevas herramientas de programación, así como de todo aquello que aparece en el mercado y os interesa conocer.

10 Programación Multimedia
PROGRAMACIÓN DE DIRECTX CON DELPHI (VII)

Siguiendo con esta serie aplicaremos lo que ya aprendimos en las anteriores entregas, creando un programa que nos dé la posibilidad de confeccionar una aplicación que nos permita la conexión de dos ordenadores en tiempo real, así podremos jugar mediante DirectPlay.

16 Redes Locales
EL ARRANQUE DE WINDOWS NT

Con este artículo veremos la secuencia de arranque de Windows NT, su comportamiento, y cómo adaptarla siempre a nuestras propias necesidades. Conoceremos más a fondo el archivo de Registro de Windows y todas sus posibilidades.

23 C/C++
PROGRAMACIÓN BÁSICA EN WINDOWS

Comenzamos una serie de artículos que nos acercan a la programación C/C++, conoceremos todos los conceptos básicos relativos al sistema operativo, y las explicaciones más elementales.

36 Linux
SVGALIB (I): UNA JOYA POR DESCUBRIR

En esta ocasión el artículo de portada nos arrastra hasta un sistema infrautilizado en lo que al tema de juegos se refiere: Linux. Pero gracias al conocimiento de una librería gráfica como puede ser SVGALIB, veremos las capacidades con las que cuenta Linux, y cómo podemos orientar este sistema hacia la creación de juegos. Una nueva posibilidad por explotar en este terreno.

29
Visual Basic
CREACIÓN DE CONTROLES ACTIVEX (y IV)

Damos por finalizada con este artículo la creación de controles ActiveX, cubriendo opciones como la de establecer una propiedad como predeterminada; agrupar las propiedades en diferentes categorías, o crear propiedades sólo accesibles en tiempo de diseño. Con ello habremos visto los principales rasgos de la generación de controles ActiveX.

44
Java
PROGRAMACIÓN DE SERVLETS (III)

Esta tercera entrega nos pondrá al corriente de la relación entre Servlets y JDBC. Los Servlets se han convertido hoy en día en una parte esencial de las arquitecturas cliente-servidor de tres capas en el entorno WWW, a lo largo del artículo analizaremos este tema.

51
Programación de sockets
INTEGRACIÓN DE CGI EN EL SERVIDOR WEB (III)

Un servidor web no tiene por qué dedicarse exclusivamente a enviar datos, sino que también puede recibirlos. Aprenderemos a ampliar el diálogo entre servidor y cliente, de forma que el segundo, también pueda enviar datos al primero.

65
Programación Hardware
PROGRAMACIÓN DEL MODEM (y IV)

Esta cuarta entrega finaliza la programación del módem abordando con especial interés el proyecto Terminal, una vez vistos en otros artículos los formularios y módulos del proyecto.

72
Libros
MANUALES PARA TODOS LOS NIVELES

Porque los libros de informática no tienen por qué ser aburridos, en esta sección os presentamos un ejemplar muy divertido para los más "torpes", y otros para aquellos que teneis más nivel en el campo de la programación.

57
Programación científica
REPRESENTACIÓN GRÁFICA DE FUNCIONES TRIDIMENSIONALES

Cómo crear programas capaces de procesar y representar gráficamente funciones tridimensionales. Nos adentraremos en el entorno gráfico donde nuestras aplicaciones podrán ser visualizadas gracias a la programación avanzada de este artículo. Tendremos la oportunidad de acceder al mundo tridimensional desde nuestro ordenador personal.

74
CORREO DEL LECTOR

En esta sección se pretende dar respuesta a todas las dudas que puedan tener nuestros lectores, pudiendo dirigirse a los propios autores o bien a la siguiente dirección de correo de nuestra publicación: solop@towercom.es.

78
CONTENIDO DEL CD-ROM

Como siempre un nuevo CD-ROM que acompaña a la revista; en esta ocasión vais a poder encontrar programas como Moray v3.0 o Xara 3D entre otros. Todos ellos servirán de complemento, junto a nuevas herramientas de programación y aplicaciones a redes, a los artículos que hemos preparado este mes en Sólo Programadores.

Noticias

También la versión de desarrollo de Internet Explorer 5.0

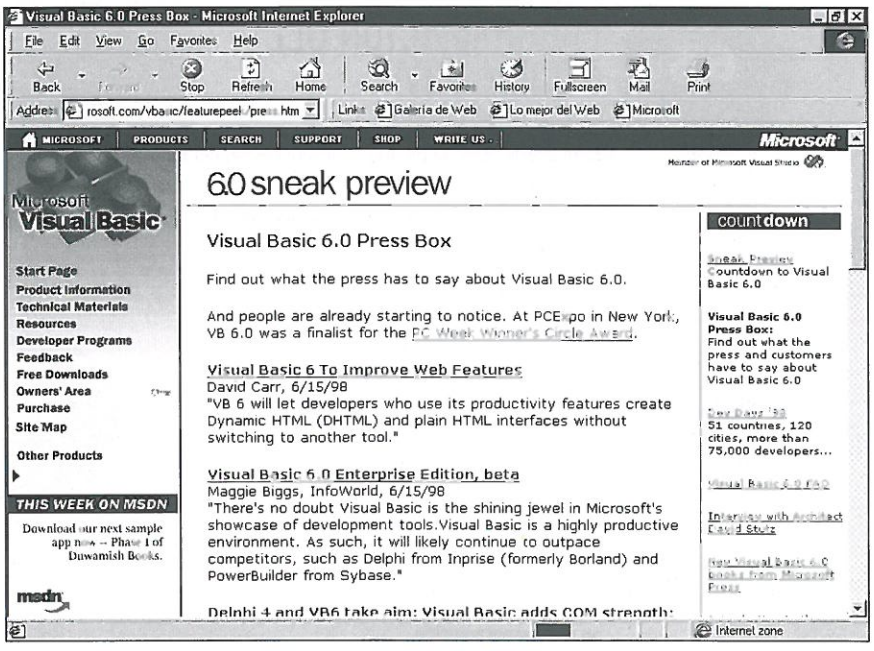
HAY ANUNCIO DE VISUAL BASIC 6.0

Esta nueva versión incluirá herramientas adicionales para hacer más fácil la construcción de servers-side, links hacia bases de datos y la gestión de otras fuentes de información a través de ActiveX Data Object (ADO). También se han añadido nuevas características que permitirán utilizar el mismo interface desde el servidor o desde el cliente. Se puede acceder a más información técnica en la siguiente dirección www.microsoft.com/vbasic/featurepeek/press.htm.

Lo que aún no se ha dado a conocer es el precio de esta nueva versión. El lanzamiento del producto está previsto para el próximo mes de septiembre y ya es seguro que formará parte del paquete Visual Studio 6.0.

Microsoft lo anunció en la conferencia TechEd de Nueva Orleans. Según fuentes de la multinacional, Visual Studio 6.0 Enterprise Edition será nuevo paquete de herramientas de desarrollo más sencillo de utilizar y más agradable para el desarrollador. Este paquete inclurá la versión 6.0 de Visual Basic, Visual C++, Visual J++, Visual InterDev, Visual FoxPro y Visual SourceSafe.

La versión de desarrollo del próximo Internet Explorer de Microsoft también ha visto la luz a mediados del pasado mes de junio. La primera copia de la versión 5.0 que usaremos se encuentra disponible en el apartado de desarrolladores de su página web principal www.microsoft.com/sitebuilder. Se trata de una release orientada únicamente a desarrolladores y hasta el momento no se considera del todo estable, por lo que no es recomendable utilizarla en equipos que no sean para análisis o experimentación ya que puede desestabilizar el sistema. Hasta el momento tan sólo hay versión disponible para plataformas de 32 bits (Windows 95, Windows 98 y Windows NT) y aún no hay anuncio oficial por parte de Microsoft sobre los planes de periodo de testeo ni de la aparición de la versión oficial.



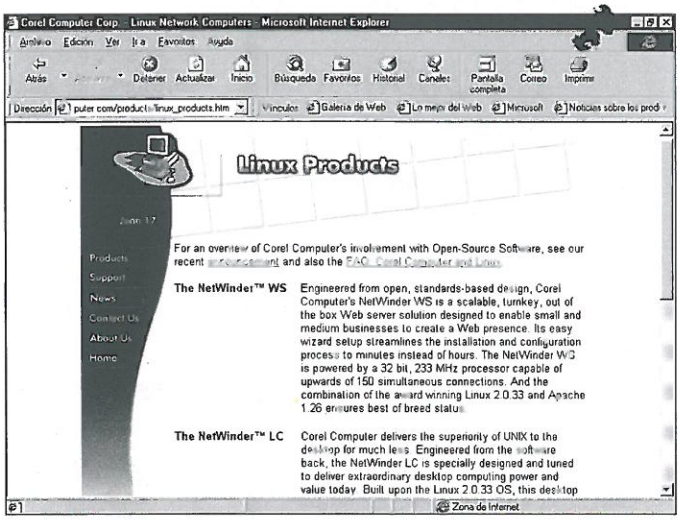
Una de las razones es la gran demanda de los usuarios
APOYO DE COREL A LA PLATAFORMA LINUX

La compañía Corel Corporation amplía su compromiso de apoyo y desarrollo al sistema operativo Linux, utilizado actualmente por unos siete millones de personas de todo el mundo. El creador del galardonado software lanzará al mercado las ediciones para PC y servidor de Corel WordPerfect 8 para Linux durante este verano. Además tiene previsto desarrollar una suite de aplicaciones empresariales para la plataforma Linux.

El apoyo a Linux concuerda perfectamente con la declaración de metas de Corel entre las que se encuentran orientar su liderazgo tecnológico para ofrecer a los clientes una excelente relación calidad-precio, compatibilidad y elección.

Actualmente Corel WordPerfect está disponible para numerosos sistemas operativos, incluyendo Linux, Unix, Macintosh, DOS y varias plataformas Windows.

Este apoyo también está en la línea de la labor de desarrollo que está realizando el subsidiario de Corel, Corel Computer Corp., en su familia de productos NetWinder. Un miembro más de la familia NetWinder es el PC basado en Linux, una terminal de trabajo del sistema Linux totalmente funcional.



Al seleccionar a Linux como su sistema operativo Corel Computer está ofreciendo a sus clientes alternativas eficaces en función de los costes y una excelente relación calidad-precio. Para obtener más información visite la siguiente dirección: **www.corelcomputer.com**

SOFTWARE AG INCLUYE MEJORAS EN SU SOFTWARE EN LA VERSIÓN 10 DE SU BASE DE DATOS RELACIONAL ADABAS D

La compañía Software AG anunció la disponibilidad de la versión 10 del sistema de bases de datos relacionales ADABAS D que ofrece numerosas mejoras con respecto a la versión anterior. El nuevo interface JDBC ofrece soporte SQL desde programas Java y proporciona acceso directo desde navegadores Web. Además ADABAS D se posiciona ahora como un servidor para trabajo en grupo para plataformas Unix y Windows NT. Diversos estudios realizados por los principales analistas indican que ADABAS D tiene el coste de propiedad más bajo entre todas las bases de datos relacionales.

El interface JDBC es Java puro cien por cien. El nivel funcional que presenta WebDB, que ha sido ampliado con los interfaces CGI, ISAPI y NSAPI, ofrece un sistema de ficheros virtual para que los usuarios de Internet almacenen sus datos y programas. También es posible realizar un acceso HTML dinámico y consultas SQL desde la Web. Por último, destacar la disponibilidad de un API para el lenguaje Perl.

Un nuevo interface gráfico de usuario está disponible para que los administradores puedan monitorizar los servidores ADABAS D desde ubicaciones remotas, incluso vía Internet, lo que elimina la necesidad de administradores in situ.

ADABAS D está optimizado para una alta disponibilidad, es un producto fácil de instalar y se ejecuta virtualmente sin la intervención del operador. El rendimiento ha sido mejorado significativamente a través de un optimizador de alta eficiencia, una organización de almacenamiento secundaria B*trees, arquitectura multihebra para la optimización de procesos paralelos y multiprocesos.

ADABAS D es comercializada a través de partners y distribuidores. La versión 10 ha sido nuevamente empaquetada y suministrado con un código EAN. Los usuarios de Microsoft pueden acceder directamente a esta base de datos, ya que ésto se ha simplificado a través del soporte ADABAS DUpsizing Wizard.

INFORMATION BUILDERS SE PREPARA PARA EL 2000

Information Builders ofrece a sus actuales y potenciales clientes un método que preparará los sistemas de las empresas para compatibilizar sus programas con el año 2000.

La entrada del nuevo siglo supone un grave y costoso problema, ya que todos los programas que usen dos dígitos para marcar el año de la fecha, no podrán diferenciar el 1900 y el 2000.

Este problema queda resuelto con las nuevas versiones de FOCUS, productos centrados en el desarrollo de aplicaciones, generación de informes y análisis de información utilizados por cientos de corporaciones, universidades y organismos de la Administración, que solucionan la cuestión del cambio de siglo con el uso de cuatro dígitos en las fechas.

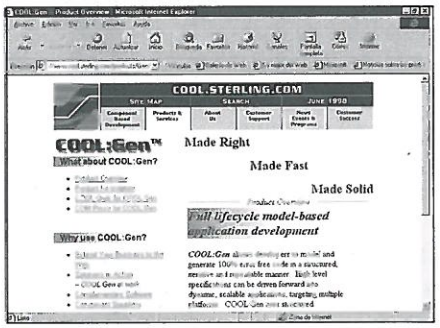
La familia FOCUS conforma una gama completa de productos, que incluye soluciones para aplicaciones empresariales, data warehousing, OLAP, herramientas de análisis y generación y publicación de datos a través de la página web.

Information Builders ofrece también Site Analyzer, una herramienta diseñada especialmente para detectar los errores relacionados con las fechas ante la llegada del año 2000, tanto en las aplicaciones de FOCUS, como en cualquier base de datos a la que tenga acceso FOCUS.

STERLING SOFTWARE CELEBRA LOS ÉXITOS DE COOL:GEN

La compañía Sterling Software líder en el desarrollo de productos software y servicios de soluciones corporativas, a través de su División de Desarrollo de aplicaciones, celebra los 11 años de desarrollo basado en modelo con su producto estrella COOL:Gen.

La herramienta COOL:Gem continúa siendo reconocida como el entorno de desarrollo de aplicaciones para empresas líder del mercado. Sterling Software en su paso hacia la nueva era de desarrollo de aplicaciones de empresa, ofrece un completo catálogo de productos integrados que están basados en modelo, incluyendo el modelado de negocios y datos, el análisis y diseño orientado a objeto y las herramientas de modelado de interface de componentes.



Con el acreditado éxito que celebran todos los clientes de Sterling como la buena calidad de los productos COOL, la compañía está planeando, en este momento, ampliar su desarrollo basado en modelo con el paradigma de la generación de códigos dentro del mundo de Java, para este próximo año.

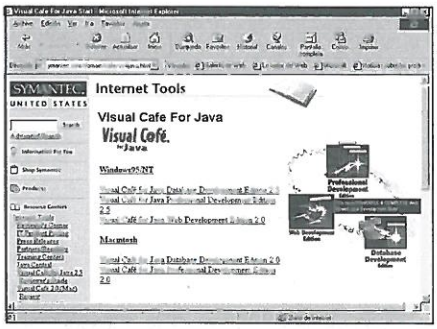
Esta compañía tiene una página web en la que puede ampliar la información que desee **www.sterling.com**

NUEVA VERSIÓN DE VISUAL CAFÉ

Symantec, compañía líder en herramientas de desarrollo de Internet, acaba de lanzar la versión 2.5 de Visual Café para Java, Edición de Desarrollo Profesional y Edición para el Desarrollo de Base de Datos.

La Edición para el Desarrollo Profesional es la herramienta más potente para la creación de aplicaciones Java, applets de Java y JavaBeans. Incluye una pantalla de usuario customizable, la posibilidad de desactivar el generador automático de código, compatibilidad para el año 2000 y un Open API. Todo ello hace que Visual Café para Java sea el más flexible y extendible de la tercera generación de las herramientas de desarrollo disponibles.

La Edición para el desarrollo de Bases de Datos, es la solución completa de desarrollo de base de datos y Java que existe en el mercado. Dirigida al profesional que desea crear, editar y utilizar el código Java para aplicaciones con conectividad a bases de datos. Ofrece lo último en actuación, flexibilidad y facilidad de uso, haciendo de ello una herramienta de elección para profesionales de Java. Los precios recomendados de venta al público son de 134.840 pesetas más IVA para la Edición de Desarrollo de Bases de Datos y de 48.220 pesetas más IVA para la Edición Profesional. Si desea ampliar la información sobre estos productos puede consultar en **www.symantec.com**



TUTOR: UN KIT DE DESARROLLO PARA DOCUMENTACIÓN DE PROCESOS Y FORMACIÓN DE USUARIOS FINALES

Oracle ha anunciado la disponibilidad en Europa de Tutor, un kit de desarrollo para documentación de procesos de usuarios finales. Tutor reduce el coste de desarrollar documentación de procesos comerciales y manuales de formación para puestos de trabajo específicos. Además incluye contenidos sobre modelos de normas y procedimientos comerciales en un formato Web que se puede integrar con los productos Oracle.

Tutor se nos presenta como una herramienta de ayuda para los directores de línea o de gestión directa de aquellas empresas que están desarrollando y desplegando aplicaciones Oracle Financials y Oracle Manufacturing.El producto está formado por tres componentes, una herramienta de autoría, una herramienta de publicaciones y un repositorio de objetos de cursos sobre aplicaciones Oracle, con documentación sobre procesos comerciales.

Todos los modelos de procesos y documentación de formación pueden ser fácilmente actualizados, mantenidos y distribuidos mediante las herramientas de autoría y de publicación, ya que cada cliente opera y maneja sus aplicaciones de forma distinta. Y se pueden crear nuevos contenidos utilizando la herramienta de autoría y las plantillas.Tutor contiene un repositorio online de más de 2000 objetos de documentos sobre cursos y procedimientos, y soporta 11 de las aplicaciones básicas de fabricación y finanzas Oracle.

La compra de Tutor incluye dos semanas de clases intensivas para enseñar a los usuarios cómo utilizar las herramientas de autoría y publicación y cómo modificar y mantener el contenido.

El precio de esta herramienta se establecerá por módulos y por la cantidad de usuarios finales, y variará según la amplitud de los procedimientos y el contenido.

IBM MEJORA LA GESTIÓN Y LA SEGURIDAD DE OS/2 Y OS/2 WARP SERVER

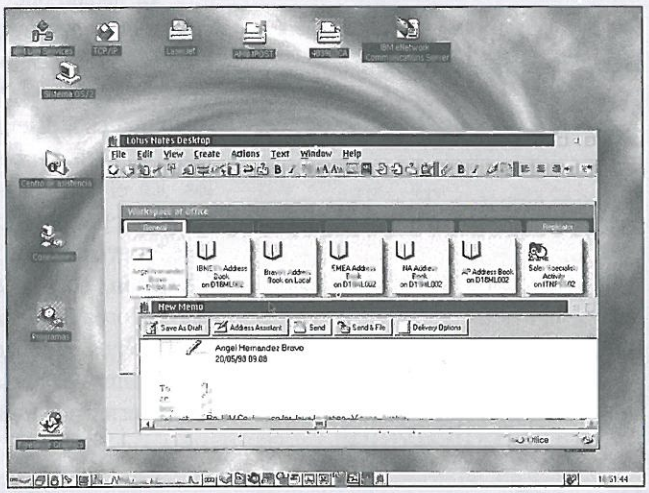
IBM ha anunciado las mejoras en las funciones de gestión del sistema, backup y recuperación de archivos del sistema operativo de red OS/2 Warp Server.

Esta actualización ofrece a los administradores la posibilidad de gestionar clientes Windows NT 4.0, al mismo tiempo que se amplía el soporte para dispositivos de hardware. OS/2 Warp Server también incluye funcionalidades nuevas, entre las que figuran la monitorización continua de los recursos y la generación de informes por horas o por meses de la utilización de los servidores.

Este sistema operativo cuenta con funciones de seguridad mejoradas, bien a través de una interfaz gráfica de usuario fácil de utilizar o por medio de una nueva interfaz de línea de comandos. Esto permite a los usuarios realizar copias de seguridad con una intervención mínima, así como recuperar los ficheros perdidos en cuestión de minutos, gracias a una función de indexado. La interfaz de línea de comandos proporciona a los administradores de red flexibilidad para automatizar y personalizar los procesos de realización de copias de seguridad en grandes redes.

Además, estos administradores pueden realizar ahora copias de seguridad en los datos en uno o varios dispositivos de backup diferentes. De esta forma los dispositivos de cinta pueden ser utilizados para recuperación de desastres, mientras que las unidades removibles como los sistemas Jaz y Zip de Iomega pueden ser instalados de forma local para recuperaciones rápidas. Estas y otras mejoras están disponibles a través de módulos que se pueden descargar en:

http://www.service.boulder.ibm/com/asd-bin/doc/es_es/home.htm



PROGRAMACIÓN MULTIMEDIA

Programación de DirectX con Delphi (VII)

Constantino Sánchez Ballesteros (constantino@redestb.es)

Para completar el estudio que hicimos en el artículo anterior sobre las posibilidades de DirectPlay y su arquitectura, vamos a crear un programa que permita la conexión física de dos ordenadores para interactuar entre sí en tiempo real.
El programa Tic Tac Toe nos permitirá jugar en Red con otros jugadores mediante DirectPlay.

INTRODUCCIÓN

DirectPlay no es uno de los más complicados de DirectX, aunque encierra un halo de confusión si no tenemos claros los conceptos de trabajo en Red, conexión entre equipos, etc. Debido al boom de Internet y sus posibilidades, se han incrementado las aplicaciones que utilizan los recursos de la Red para establecer conexiones entre múltiples usuarios que comparten una misma aplicación.

DirectPlay no sólo está orientado a juegos, sino que podemos crear programas tipo Chat (muy comunes en Internet para hablar con otras personas), bases de datos, etc., para transferir datos de un lugar a otro y de forma transparente para el usuario.

El listado que veremos a continuación representa un programa del tipo "3 en raya" que permite la conexión simultánea de 2 jugadores. Por supuesto, como el programa está diseñado para la conexión con otra persona, no podremos jugar solos. En el momento en que se carga el programa por primera vez, si seleccionamos **Nueva** desde el menú del formulario **Conexión**, podremos elegir uno de los proveedores que permite *DirectPlay*. Recordemos que mediante los proveedores podemos conectarnos físicamente a una determinada Red.

Una vez que seleccionamos el proveedor, (ya sea *TCP/IP*, *IPX*, etc.), esperaremos a que el otro jugador se conecte con nosotros para iniciar la sesión. En en el momento en el que la conexión esté establecida, podremos jugar sin ningún tipo de limitaciones.

TIC TAC TOE

En la Figura 1 podéis apreciar un detalle del formulario que contiene la parte visual del programa.

En el formulario aparece un componente de tipo menú para la gestión de comandos y otro componente de tipo *DGCPlay* para establecer la interfaz de *DirectPlay* en el programa. Como comenté anteriormente, debemos elegir la opción **Nueva** del menú **Conexión** para poder establecer una conexión con otro equipo. Después de este paso, sólo tendremos que seleccionar la opción **Unirse** cuando se haya detectado al otro jugador, y así poder comenzar la partida.

Podemos elegir conexión a Internet, por cable, módem a módem y puerto serie. Por supuesto, debemos tener configurado Windows 95 para cada tipo de conexión que queramos establecer. A continuación damos paso al listado del programa:

```
unit Unit1;

interface
```

```
uses
  Windows, Messages, SysUtils, Classes, Graphics,
      Controls, Forms, Dialogs,
  StdCtrls, DGCPlay, ExtCtrls, Menus, DPLAY,
      OLE2, Buttons;
```

Para la gestión de las acciones de los jugadores, se definen 2 constantes: *TTT_MOVEMSG* para cuando se efectúa un movimiento por parte de algún jugador y *TTT_NEWMSG* para comenzar un nuevo juego.

```
const
  TTT_MOVEMSG = 20;
  TTT_NEWMSG  = 21;

type
  TTT_MOVE = record
    dwType : DWORD;
    Square : Integer;
  end;
  TTT_NEWGAME = record
    dwType : DWORD;
end;
```

La clase **Tform** es la que representa al formulario y contiene los botones y menú que vamos a utilizar en el programa:

```
  TForm1 = class(TForm)
    DPlay: TDGCPlay;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    MainMenu1: TMainMenu;
    Game1: TMenuItem;
    New1: TMenuItem;
    Join1: TMenuItem;
    End1: TMenuItem;
    N1: TMenuItem;
    Exit1: TMenuItem;
    Button1: TBitBtn;
    Button2: TBitBtn;
    Button3: TBitBtn;
    Button4: TBitBtn;
    Button5: TBitBtn;
    Button6: TBitBtn;
    Button7: TBitBtn;
    Button8: TBitBtn;
    Button9: TBitBtn;
    Game2: TMenuItem;
    NewGame1: TMenuItem;
```

Definimos los procedimientos que utilizaremos en el programa:

```
    procedure Exit1Click(Sender: TObject);
    procedure New1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Join1Click(Sender: TObject);
    procedure DPlayAddPlayer(ID: Integer;
        LongName, ShortName: string);
    procedure Button1Click(Sender: TObject);
    procedure DPlayGameMessage(From, MsgType:
        Integer; GameMsg: Pointer);
    procedure NewGame1Click(Sender: TObject);
    procedure End1Click(Sender: TObject);
    procedure DPlayDeletePlayer(ID: Integer;
        LongName, ShortName: string);

private
    Procedure DrawAllSquares;
    Procedure ClearAllSquares;
    function  CheckForWinner : Integer;
    Procedure WeHaveAWinner(Line : Integer);

var
  Form1: TForm1;
```

Para poder establecernos correctamente en la Red que determinemos, es necesario registrarnos mediante la ayuda de un *GUID*. Recordemos que un *GUID* es un identificador global que se define en el registro de Windows, y tiene como misión dar constancia de que un determinado dispositivo existe. ¡¡¡Sin este GUID no podremos identificarnos en la Red!!!.

```
  MyGameGUID   : TGUID =
      (D1:$C936BB40;D2:$5A82;D3:$11D0;D4:($
      BA,$6C,$00,$A0,$24,$D0,$D1,$DF));

  MyTurn:          Boolean;
  RemotePlayer:    DPID;
  IAmHost:         Boolean;
  MoveMessage:     TTT_MOVE;
  Square:          Array[1..9] of Char;
  Winner:          Char;
  GameOver:        Boolean;

implementation

uses Unit2;
```

Establecemos nuestra identificación (ID) en la Red mediante la ayuda del GUID definido anteriormente. Al efectuar esta operación, seremos los únicos que tengamos estos datos en la Red, es decir, que no puede haber 2 equipos con el mismo ID. Además, preparamos los botones que definirán las casillas del juego borrando su contenido mediante el procedimiento *ClearAllSquares*:

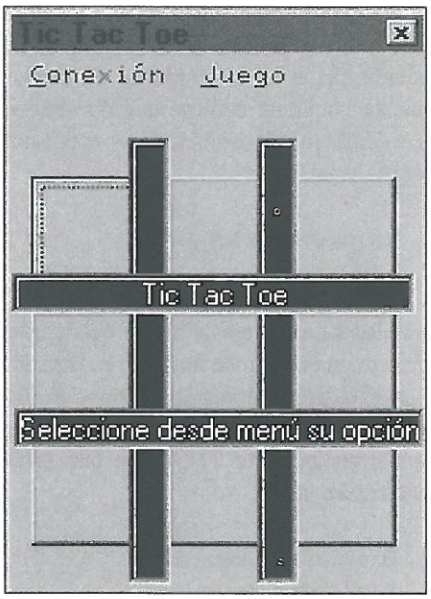


Figura 1. Formulario de Tic Tac Toe.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
   End1.Enabled := False;
   DPlay.GameGUID  := MyGameGUID;
   ClearAllSquares;
end;
```

Cuando se selecciona la opción de menú **Salir**, se ejecuta este procedimiento que cierra el formulario y termina la aplicación:

```
procedure TForm1.Exit1Click(Sender: TObject);
begin
   Close;
end;
```

Si se selecciona un nuevo juego desde el menú de formulario, se ejecuta automáticamente este procedimiento. En él se detectan los proveedores que tenemos instalados en Windows para su posterior selección:

```
procedure TForm1.New1Click(Sender: TObject);
begin
```

Lo primero que hacemos es establecer nuestro nombre de jugador para la identificación en el programa. Es conveniente poner un nombre que no utilicen los demás jugadores para evitar confusión entre éstos:

```
DPlay.ShortName := 'Jugador 1';
```

Asignamos los nombres de los proveedores en la "caja de listas" que hemos puesto en el segundo formulario. Para obtener los proveedores de servicios es necesario utilizar el método *ProviderNames* en el componente *TDGCPlay* que estamos utilizando:

```
Form2.ListBox1.Clear;
Form2.ListBox1.Items :=
   Dplay.ProviderNames;

Form2.Caption := 'Seleccione un proveedor de
   servicios';
Form2.ShowModal;
If Form2.ListBox1.ItemIndex < 0 then exit;
```

Una vez que el usuario ha seleccionado el proveedor adecuado, lo abrimos mediante el método *OpenProvider* y esperamos la conexión con el otro jugador.

```
With Form2.ListBox1 do

      DPlay.OpenProvider(Items[ItemIndex],
   True);

   New1.Enabled := False;
   Join1.Enabled := False;
   End1.Enabled := True;
   Panel4.Caption := 'Esperando al Jugador 2 para
      conectar';
   IAmHost := True;
end;
```

El siguiente procedimiento se encarga de unir a ambos contendientes en el juego. Debemos iniciar la sesión en el proveedor seleccionado y elegir el jugador que se ha conectado con nosotros:

```
procedure TForm1.Join1Click(Sender: TObject);
begin
```

El jugador que se conecte a nosotros llevará el nombre de "jugador 2"...

```
   DPlay.ShortName := 'Jugador 2';
   Form2.ListBox1.Clear;
   Form2.ListBox1.Items :=
      DPlay.ProviderNames;

   Form2.Caption := 'Elija un proveedor';
   Form2.ShowModal;
   If Form2.ListBox1.ItemIndex < 0 then Exit;

   With Form2.ListBox1 do
      DPlay.OpenProvider(Items[ItemIndex],
      False);

   New1.Enabled    := False;
   Join1.Enabled   := False;
   End1.Enabled    := True;
   NewGame1.Enabled := False;
```

Para saber si se ha establecido correctamente la sesión entre ambos jugadores es necesario consultar al método *NumberOfSessions*. Si el número obtenido es menor de 1 significa que no se estableció ninguna sesión con ningún otro equipo en la Red.

```
   If DPlay.NumberOfSessions < 1 then
   begin
     ShowMessage('No se establecieron sesiones
       en este proveedor');
     exit;
   end;
```

Por el contrario, si se encontró alguna sesión, se visualizará en la "caja de listas" del formulario 2 mediante el método *Sessions*. El siguiente paso será seleccionar por parte de los usuarios la sesión detectada. Una vez cumplido este requisito, se abrirá dicha sesión mediante el método *OpenSession*:

```
   Form2.Caption := 'Seleccione una sesión para
      el juego';
   Form2.ListBox1.Clear;
   Form2.ListBox1.Items := DPlay.Sessions;
   Form2.ShowModal;

   If Form2.ListBox1.ItemIndex < 0 then exit;
   With Form2.ListBox1 do

      DPlay.OpenSession(dword(Items.Objects[Ite
      mIndex]));

   With DPlay.PlayerSNames do
      RemotePlayer :=
        dword(Objects[IndexOf('Jugador 1')]);

   IAmHost := False;
   MyTurn  := False;
   Panel4.Caption := 'Esperando movimiento del
      jugador 1';
end;
```

Si decidimos acabar la partida, debemos cerrar la sesión activa y reiniciar los valores para comenzar de nuevo el juego. Mediante el método *CloseSession* se cierra la sesión desechando los valores establecidos al inicio del juego.

```
procedure TForm1.End1Click(Sender: TObject);
begin
   DPlay.CloseSession;
   ClearAllSquares;
   DrawAllSquares;
   Panel4.Caption := 'Esperando la selección del
      menú';
   New1.Enabled    := True;
   Join1.Enabled   := True;
   End1.Enabled    := False;
   NewGame1.Enabled := False;
end;
```

Cuando se ha detectado un nuevo jugador en la conexión, enviamos un mensaje en pantalla indicándolo y se le invita a realizar su primer movimiento en el juego que estamos desarrollando:

```
procedure TForm1.DPlayAddPlayer(ID: Integer;
      LongName, ShortName: string);
begin
   Panel4.Caption := 'Jugador 2 conectado!';
   Sleep(1000);
   Panel4.Caption := 'Tú mueves! Elige un cua-
      drado';
   NewGame1.Enabled := True;
   MyTurn := True;
end;
```

En el caso de que el otro jugador haya cortado la conexión con nuestro ordenador, debemos reiniciar el panel de las casillas representado por botones para poder comenzar un nuevo juego cuando sea necesario:

```
procedure TForm1.DPlayDeletePlayer(ID: Integer;
      LongName, ShortName: string);
```

```
begin
   ClearAllSquares;
   DrawAllSquares;
   if IAmHost then
     Panel4.Caption := 'Jugador 2 salió!'
   else
     Panel4.Caption := 'Jugador 1 salió!';
end;
```

El siguiente procedimiento se encarga de poner sobre las casillas (botones del formulario) los caracteres **X** u **O** para gestionar los movimientos realizados por los jugadores. Si nosotros somos el *Host* de la Red, utilizaremos **O**.

```
procedure TForm1.DPlayGameMessage(From,
      MsgType: Integer; GameMsg: Pointer);
var
   WhatPlayer : Char;
Begin
```

// Si el mensaje que hemos recibido del otro jugador es de movimiento de fichas (**TTT_MOVEMSG**)...

```
 Case MsgType of
 TTT_MOVEMSG:
  begin
  If IAmHost then WhatPlayer := 'O' else
      WhatPlayer := 'X';
      Square[TTT_MOVE(GameMsg^).Square]
      := WhatPlayer;
      MyTurn := True;
      Panel4.Caption := 'Tú mueves!';
      DrawAllSquares;
  end;
```

// Si no, significa que estableceremos un nuevo juego entre los jugadores.

```
 TTT_NEWMSG:
    Form1.NewGame1Click(Self);
   end;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
```

Antes de poner **X** u **O** sobre una casilla, debemos comprobar que el juego no ha concluido, que no pulsamos sobre una casilla que ya tiene un movimiento realizado y si es nuestro turno de movimiento.

```
   If GameOver = True then exit;
   If Square[ (Sender as TButton).Tag ] <> '' then
      exit;
   If Not MyTurn then exit;

   If IAmHost then
     Square[(Sender as TButton).Tag] := 'X'
   else
     Square[(Sender as TButton).Tag] := 'O';
```

Una vez que realicemos el movimiento (selección de una casilla), debemos enviar un mensaje de movimiento (**TTT_MOVEMSG**) al otro jugador, es decir, le cedemos el turno. Para ello, necesitamos utilizar un método llamado *SendMessage* en el componente TDGCPlay. La variable *IAmHost* determina si somos el Host de la conexión y *MyTurn* define si es nuestro turno en la jugada.

```
   MoveMessage.dwType := TTT_MOVEMSG;
   MoveMessage.Square := (Sender as
      TButton).Tag;
   DPlay.SendMessage(RemotePlayer,
      MoveMessage);
   If IAmHost then
     Panel4.Caption := 'Esperando el movimiento
      del jugador 2'
   else
     Panel4.Caption := 'Esperando el movimiento
      del jugador 1';

   DrawAllSquares;
   MyTurn := False;
end;
```

El siguiente procedimiento se encarga de redibujar las casillas con los valores establecidos en el juego (**X-O**) y chequea si hay un ganador que haya conseguido hacer línea.

```
Procedure TForm1.DrawAllSquares;
var
  Line : Integer;
begin
  Button1.Caption := Square[1];
  Button2.Caption := Square[2];
  Button3.Caption := Square[3];
  Button4.Caption := Square[4];
  Button5.Caption := Square[5];
  Button6.Caption := Square[6];
  Button7.Caption := Square[7];
  Button8.Caption := Square[8];
  Button9.Caption := Square[9];
```

La variable *Line* obtendrá un valor del procedimiento *CheckForWinner* para saber si hay un vencedor en la partida. Si este valor es mayor que 0, significa que lo hay.

```
  Line := CheckForWinner;
  If Line > 0 then WeHaveAWinner(Line);
end;
```

// Borra el contenido de las casillas cuando sea necesario

```
Procedure TForm1.ClearAllSquares;
var
   Loop : Integer;
begin
   For Loop := 1 to 9 do
      Square[Loop] := #0;
   DrawAllSquares;
   GameOver := False;
end;
```

// Busca un ganador en la partida ...

```
function TForm1.CheckForWinner : Integer;
Var
   Loop : Integer;
   C    : Char;
   Tie  : Boolean;
begin
   For Loop := 1 to 2 do
   begin
    If Loop = 1 then
      C := 'X'
    else
      C := 'O';
```

Para saber si se ha establecido una línea con los mismos caracteres, debemos ir casilla por casilla (representadas por la variable *Square[]*) y comprobar que sus tres caracteres contienen los mismos valores; **X** u **O**. Es necesario establecer todas las posibilidades de línea que se puedan dar con las nueve casillas del panel.

```
    if Square[1] + Square[2] + Square[3] =
      C+C+C then
      begin
        Result := 1;
        Winner := C;
        exit;
      end;
    if Square[4] + Square[5] + Square[6] =
      C+C+C then
      begin
        Result := 2;
```

```
    Winner := C;
    exit;
   end;
 If Square[7] + Square[8] + Square[9] =
   C+C+C then
   begin
    Result := 3;
    Winner := C;
    exit;
   end;
 If Square[1] + Square[4] + Square[7] =
   C+C+C then
   begin
    Result := 4;
    Winner := C;
    exit;
   end;
 If Square[2] + Square[5] + Square[8] =
   C+C+C then
   begin
    Result := 5;
    Winner := C;
    exit;
   end;
 If Square[3] + Square[6] + Square[9] =
   C+C+C then
   Begin
    Result := 6;
    Winner := C;
    exit;
   end;
 If Square[1] + Square[5] + Square[9] =
   C+C+C then
   Begin
    Result := 7;
    Winner := C;
    exit;
   end;
 If Square[3] + Square[5] + Square[7] =
   C+C+C then
   begin
    Result := 8;
    Winner := C;
    exit;
   end;
end;
Result := 0;
Tie := True;
For Loop := 1 to 9 do
  If Square[Loop] = #0 then Tie := False;
If Tie = True then
begin
 Result := 9;
 Winner := 'T';
 exit;
 end;
end;
```

// Ganador encontrado !!!

```
Procedure TForm1.WeHaveAWinner(Line :
    Integer);
begin
  Case Winner of
   'X': Panel4.Caption := 'Jugador 1 gana!';
   'O': Panel4.Caption := 'Jugador 2 gana!';
   'T': Panel4.Caption := 'Empate!!';
  end;
  GameOver := True;
```

Cuando encontramos un ganador, cambiamos el color de la línea ganadora para que sobresalte (color rojo):

```
  Case Line of
   1:
    Begin
     Button1.Font.Color := clRed;
     Button2.Font.Color := clRed;
     Button3.Font.Color := clRed;
    end;
   2:
    Begin
     Button4.Font.Color := clRed;
     Button5.Font.Color := clRed;
     Button6.Font.Color := clRed;
    end;
   3:
    Begin
     Button7.Font.Color := clRed;
     Button8.Font.Color := clRed;
     Button9.Font.Color := clRed;
    end;
   4:
    Begin
     Button1.Font.Color := clRed;
     Button4.Font.Color := clRed;
     Button7.Font.Color := clRed;
    end;
   5:
    Begin
     Button2.Font.Color := clRed;
     Button5.Font.Color := clRed;
     Button8.Font.Color := clRed;
    end;
   6:
    Begin
     Button3.Font.Color := clRed;
     Button6.Font.Color := clRed;
     Button9.Font.Color := clRed;
    end;
   7:
    Begin
     Button1.Font.Color := clRed;
     Button5.Font.Color := clRed;
     Button9.Font.Color := clRed;
    end;
   8:
    Begin
     Button3.Font.Color := clRed;
     Button5.Font.Color := clRed;
     Button7.Font.Color := clRed;
    end;
  end;
end;
```

Este procedimiento sirve para iniciar el formulario con nuevos valores cuando se selecciona un nuevo juego entre jugadores. El color de los caracteres de las casillas se establecerá en negro (cuando alguien gana se convierte la línea en color rojo):

```
procedure TForm1.NewGame1Click(Sender:
     TObject);
var
  NewGameMsg  : TTT_NEWGAME;
begin
  Button1.Font.Color := clBlack;
  Button2.Font.Color := clBlack;
  Button3.Font.Color := clBlack;
  Button4.Font.Color := clBlack;
  Button5.Font.Color := clBlack;
  Button6.Font.Color := clBlack;
  Button7.Font.Color := clBlack;
  Button8.Font.Color := clBlack;
  Button9.Font.Color := clBlack;
  ClearAllSquares;
  DrawAllSquares;
  If IAmHost = True then
  begin
    MyTurn := True;
    Panel4.Caption := 'Elige una casilla!';
    NewGameMsg.dwType := TTT_NEWMSG;
    DPlay.SendMessage(RemotePlayer,
     NewGameMsg);
  end
  else
  begin
    MyTurn := False;
    Panel4.Caption := 'Esperando el movimiento
     del jugador 1';
  end;
end;

end.
```

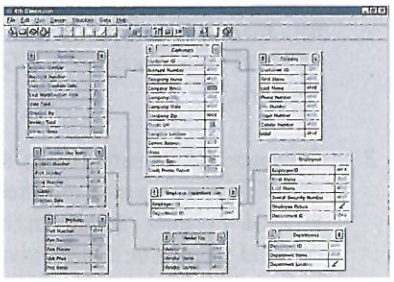
Si le gustaron otras bases de datos, 4th Dimension v6 *le sorprenderá*

4th Dimension es el entorno de bases de datos integrado más completo del mercado. Años de experiencia en el diseño de bases de datos han permitido satisfacer las siguientes necesidades:

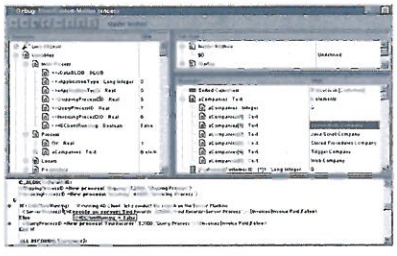
- Automatizar cualquier proceso.
- Diseñar modelos gráficos eficientes y orientados a objeto.
- Suministrar la más amplia gama de herramientas para facilitar el diseño y mantenimiento de bases de datos.
- Permitir una fácil reutilización de código.
- Permitir escalar desarrollos Cliente/Servidor, web y multiplataforma de forma transparente.

Estas son algunas de las **características** de 4th Dimension:

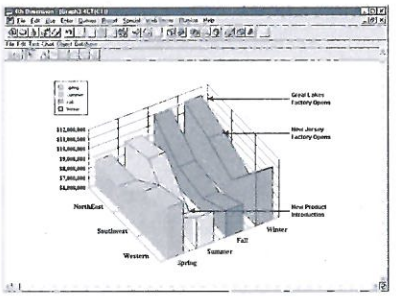
- **Motor de base de datos multiproceso, triggers, procedimientos almacenados**, presentación visual de tablas mediante diagrama de entidad /relación (ERD)



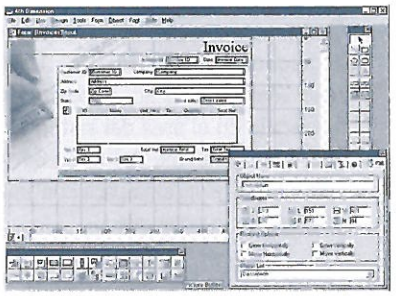
- **Debugger visual** multiproceso de última generación. **Intérprete** incluido para testeo y debugging.



- **Wizards avanzados** para diseño de formularios, queries, informes, importación, exportación, etiquetas, etc. Editor de gráficas programable 2D y 3D, soporta gráficas de barras, tartas, áreas, líneas, etc.

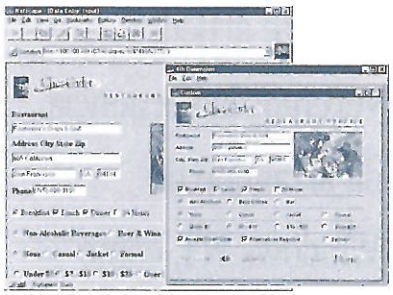


- **Editor de formularios basado en objetos.** Gestión de hojas de estilo. Redimensionamiento automático. **Docenas de objetos de interface** incluyendo check boxes, radio buttons, push buttons, áreas de scroll, picture button, botones invisibles, tab control, listas jerárquicas, termóme-tros, reglas, combo box, listas menú desplegables, menús pop-up jerárquicos, menús picture, radio pictures, listas desplegables, menús pop-up, subforms, group boxes, etc.

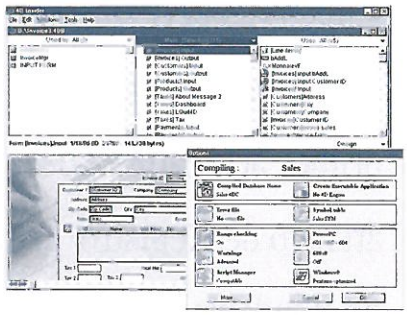


- Arquitectura de distribución de aplicaciones integrada y **soporte para proyectos con múltiples desarrolladores** con check-in/check-out automatizado.
- Arquitectura extensible mediante **plug-ins** y DLLs.

- **Un sólo código fuente** desplegable sin modificaciones de monousuario a 4D Server, la versión Cliente/Servidor de 4th Dimension.
- Soporte de **ODBC** (cliente y servidor) y conexiones nativas **SQL** a ORACLE, Sybase y MS SQL Server.
- **Servidor Web integrado** para acceso automático a formularios y gestión Cliente/Servidor mediante navegadores web.



- **Compilador a código máquina real** optimizado para diferentes procesadores. **Gestor gráfico de referencias cruzadas** y librerías de objetos.



VERSIÓN DEMO GRATUITA

4th Dimension es el núcleo de la gama de productos 4D. Visite http://www.ambitconsulting.com o llame al 93-209 41 11 para más información o para recibir una versión demo gratuita. Mencione este anuncio y reciba 4th Dimension v6 por el precio promocional de 34.900 Ptas*. (precio habitual: 79.000 Ptas.).

(* Actualización competitiva. Oferta válida hasta el final de existencias)

Balmes, 297, 4º, 2ºb. 08006 Barcelona
t. | (34) 93 209 41 11 f. | (34) 93 240 18 88
e-mail: ambit@ambitconsulting.com
http://www.ambitconsulting.com

AMBIT Consulting

El arranque de Windows NT

Enrique Díaz Trobo (enriqued@jet.es)

En este artículo veremos la secuencia de arranque de Windows NT, cómo funciona y cómo ajustarla lo más posible a nuestras necesidades. También haremos un primer acercamiento al archivo de Registro de Windows NT, y aprenderemos a realizar una conexión automática con un nombre de usuario y contraseña.

El proceso de arranque de Windows NT comprende tres fases distintas:

- La secuencia de prearranque
- La secuencia de arranque
- La secuencia de carga de Windows NT

Todos los procesos de la secuencia de prearranque están controlados por los programas que están almacenados en la memoria ROM y consta de los siguientes procesos:

- Chequeo inicial del equipo.
- Carga y ejecución del registro de arranque maestro (MBR, *Master Boot Record*). Dicho registro está localizado en el área del sistema del sector de arranque del disco duro y contiene el programa para continuar el proceso de arranque.
- Carga del sector de arranque. El MBR examina el registro de arranque de la partición (PBR *Partition Boot Record*), para identificar la partición activa del disco duro y carga el sector de arranque de dicha la partición activa.
- Carga del archivo NTLDR. Este con el que trabajamos, es un archivo oculto de sólo lectura, que se encuentra en el sector de arranque y puede arrancar Windows NT y otros sistemas operativos.

LA SECUENCIA DE ARRANQUE

Esta fase comienza cuando NTLDR comienza a cargar el sistema operativo y consta de seis procesos:

1. Cambio al modelo de memoria flat. NTLDR cambia el modo de trabajo del microprocesador de 16 a 32 bits, utilizando el modelo de memoria flat. Este modo posibilita que Windows NT pueda acceder a 4 Gb de Memoria.

2. Arranque del miniarchivo de sistema. NTLDR carga este miniarchivo, lo cual permite cargar archivos desde el sistema de archivos (ya sea FAT o NTFS) en la partición de arranque.

3. Preparación del menú de arranque. NTLDR prepara un menú de arranque, y lo hace así incluso si Windows NT es el único sistema operativo existente en el disco duro. Esto permite escoger entre cargar Windows NT normalmente o en modo VGA. Este modo de vídeo nos permite corregir diversos problemas de presentación que pudieran aparecer. El menú está definido por las entradas que existen en el archivo BOOT.INI. Si existen otros sistemas operativos en una configuración multiarranque, también aparecerán en este menú,

ya que las alternativas posibles estarán escritas en el archivo BOOT.INI.

4. Carga del sistema operativo. Después de la selección realizada por el usuario, NTLDR carga el sistema operativo y arranca ntdetect.com si es que hemos seleccionado Windows NT para arrancar. Posteriormente se continúan los procesos de arranque. Si seleccionamos otro sistema operativo, NTLDR transfiere el control a los archivos de arranque del sistema operativo seleccionado.

5. La Detección del Hardware: Ntdetect.com examina el ordenador, determina la configuración del equipo y devuelve la información a NTLDR.

6. Carga del núcleo o kernel. NTLDR arranca ntoskrln.exe (el núcleo del sistema operativo que maneja directamente los procesadores) y le pasa la información de configuración del equipo que había recibido previamente de ntdetect.com.

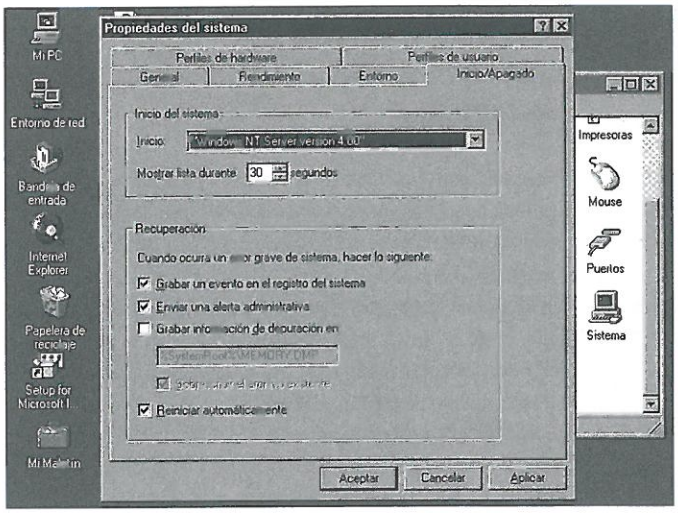


Figura 1. Inicio/Apagado nos permite modificar algunas opciones de arranque.

LA FASE DE CARGA

El control de la fase de carga la lleva a cabo ntoskrln.exe y comprende cuatro fases:

1. Continuación de la carga del núcleo o kernel. Este paso comienza cuando se ha cargado ntoskrln.exe y se producen dos procesos:

   - Se carga el nivel de abstracción del hardware (HAL).
   - El sistema de registro (una parte del registro, con el que iremos más adelante) es revisado para identificar qué unidades y servicios serán iniciados.

   Este paso de la fase de carga se produce después de limpiarse la pantalla y aparecen una serie de puntos suspensivos en la parte superior de la pantalla.

2. Inicialización del núcleo o kernel. En este paso se inicializa el núcleo y las unidades después de haber sido cargado.

   Después de que se hayan inicializado las unidades, el CurrentControlSet del Registro es guardado y se crea un conjunto de control Clone. Para realizar este paso se utiliza la información obtenida por ntdetect.com.

Durante la inicialización del núcleo se pueden producir cuatro niveles de ErrorControl:

- 0x0. Este código informa de un error *Ignore* que es ignorado por la secuencia de arranque y no se presenta ningún error en la pantalla
- 0x1. Este código informa de un error *Normal* que es ignorado por la secuencia de arranque pero que sí presenta un mensaje de error en pantalla.

El arranque de Windows NT consta de tres fases principales

- 0x2. Este código informa de un error *Severe*. Si el sistema no está siendo arrancado con la *Última configuración buena conocida*, ésta será utilizada. Si el sistema está siendo arrancado con la *Última configuración buena conocida*, el error se ignorará.
- 0x3. Este código informa de un error *Critical*. La secuencia de arranque falla y vuelve a comenzar utilizando la *Última configu-*

Listado 1. Entradas del archivo boot.ini

```
[boot loader]
timeout=30
default=C:\
[operating systems]
C:\="Microsoft Windows"
multi(0)disk(0)rdisk(0)partition(3)\WINNT="Windows NT Server versión 4.00"
multi(0)disk(0)rdisk(0)partition(3)\WINNT="Windows NT Server versión 4.00 [modo VGA]"
/basevideo /sos
```

Listado 2. Categorías principales del Registro

CATEGORÍA	DESCRIPICIÓN
HKEY_CLASSES_ROOT	Esta rama del Registro contiene archivos y vínculos OLE. En realidad no es muy necesario, pues todos los datos que aparecen aquí también se encuentran en HKEY_LOCAL_MACHINE_SOFTWARE.
HKEY_CURRENT_USER	En esta rama se muestra el perfil del usuario recién incorporado, es decir, las carpetas del usuario, los colores de pantalla, y las configuraciones del *Panel de Control*. En realidad es un subconjunto de HKEY_USERS.
HKEY_LOCAL_MACHINE	Se trata del área más importante del registro. Contiene información sobre el hardware del ordenador y el software instalado. Hay partes que se reconstruyen cada vez que se inicia el ordenador para reflejar la configuración del ordenador en cada arranque.
HKEY_USERS	Contiene todos los perfiles de los usuarios, así como el perfil por defecto. Tras el primer inicio sólo hay dos perfiles: el perfil por defecto y el perfil del usuario existente, que aparece como una serie de números precedidos por una *S* y separado por guiones.
HKEY_CURRENT_CONFIG	En esta rama están las configuraciones actuales relativas al equipo y a sus programas.

ración buena conocida. Si esta configuración ya estaba siendo utilizada, la secuencia de arranque falla y se presenta en pantalla un mensaje de error. Sólo nos queda recurrir a los disquetes de recuperación y, en último caso, reinstalar.

3. Carga de los servicios. En este paso se cargan los servicios a través de smss.exe (el *Session Manager*) siguiendo esta secuencia:

   - *Session Manager* ejecuta los programas indicados en el registro, debajo del valor BootExecute (este valor se encuentra en el registro, bajo:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SessionManager).

El valor que nos encontramos por defecto de éste es *utocheck autochk* * el cual ejecuta autochk.exe. Este archivo realiza la misma función que el comando chkdsk del DOS en todos los volúmenes y muestra información sobre cada volumen en la pantalla.

Si dicho valor se cambia a *autochek autochk /p* *, autochk.exe tendrá la misma función que el comando chkdsk /f del DOS. Es decir, no sólo detecta los errores, sino que también los corrige.

NTLDR permite cargar varios sistemas operativos

También se pueden añadir otros programas en este valor, situando cada comando en una línea separada con el Editor del Registro (más adelante iremos con ello)

- *Session Manager* configura las páginas de archivos definidas en el Registro, bajo:

HKEY_LOCAL_MACHINE\SYSTEMCurrentControlSet\Control\Session Manager\Memory Management.

- CURRENTCONTROLSET y el control CLONE se escriben en el Registro.

- *Session Manager* carga los subsistemas necesarios que están definidos en el registro bajo HKEY_LOCAL_MACHINE\SYSTEM\CURRENTCONTROLSET\CONTROL\SESSION MANAGER\SUBSYSTEMS. Por defecto, sólo está definido Win32 como subsistema requerido.

4. Iniciación del subsistema Windows. Este paso cuenta a su vez con dos procesos:

   - Se ejecuta winlogon.exe que, a su vez, ejecuta lsass.exe (Autoridad de Seguridad Local) que se encarga de mostrar la pantalla de diálogo de conexión pidiendo el nombre de usuario y la contraseña, así como el dominio al que quiere conectarse.

   - Se ejecuta screg.exe (Controlador de Servicio) que examina el registro para identificar los servicios que están configurados para ser cargados automáticamente. Los servicios son cargados en orden a sus dependencias (definidas en los valores DEPENDONGROUP y DEPENDONSERVICE)

     Los servicios están definidos en el registro bajo:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services.

Si tomamos como ejemplo, en HKEY_LOCAL_MACHINE\SYSTEM\CURRENTCONTROLSET\SERVICES\BROWSER, están los valores DependOnGroup y DependOnService indicados anteriormente en el listado expuesto.

PERSONALIZAR EL PROCESO DE ARRANQUE

El proceso de arranque se controla mediante el registro y el archivo BOOT.INI.

El archivo BOOT.INI es un archivo de texto almacenado en el directorio raíz de la unidad C: y se crea en el momento de la instalación de Windows NT. Este archivo tiene los atributos de sólo lectura y oculto.

Podemos modificar este archivo quitando los atributos de sólo lectura y sistema y editándolo con cualquier editor de textos ASCII.

En el Listado 1 tenéis un ejemplo de este archivo. Si nos fijamos en este listado, las particiones en este archivo están identificadas con las convenciones ARC (*Advanced RISC Computer*), que definen el disco duro y sus particiones.

En el caso de las unidades SCSI, los dispositivos se identifican:

```
Scsi(a) disk(b) rdisk(c) partition(d)
```

Otros dispositivos se identifican en el formato:

```
Multi(a) disk(b) rdisk(c) partition(d)
```

El significado de estas líneas es:

- scsi/multi(a) identifica el adaptador utilizado, donde (a) es el número del adaptador (no olvidemos que el cero se corresponde con el primer adaptador, uno con el segundo y así sucesivamente)
- disk(b) identifica la dirección del dispositivo en el bus SCSI, si utilizamos el parámetro multi, el identificativo (b) siempre ha de ser cero.
- rdisk(c) identifica el número de disco cuando se utiliza el parámetro multi, siendo el primer disco cero. Si utilizamos el parámetro scsi, siempre será cero.
- partition(d) identifica la partición en la unidad especificada en el parámetro anterior. En este caso la primera partición se corresponde con un uno.

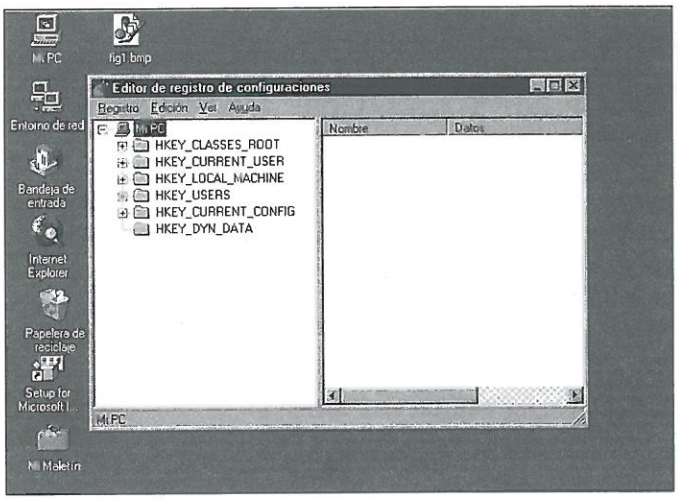


Figura 2.
El Editor del registro nos permite acceder a los valores del sistema.

EL ARCHIVO BOOT.INI

Este archivo tiene la estructura de los archivos .INI de toda la vida. Así pues, tendremos en él secciones; en este caso [boot loader] y [operating system].

La sección *boot loader* establece el sistema operativo por defecto que se cargará si no se realiza una selección por parte del usuario. En esta sección podemos establecer dos valores:

- *Timeout=n* Este valor establece un intervalo de tiempo de *n* segundos durante el cual se espera una selección del usuario. Si no se realiza ninguna selección se arrancará con el sistema operativo predeterminado.
- *Default=partición/directorio*. Este valor establece la partición (en formato ARC) y el directorio del sistema operativo predeterminado; el que se cargará cuando transcurra el lapso de tiempo indicado en *timeout*.

Una forma segura de modificar estas opciones es mediante la utilidad que aparece si pulsamos el botón *Inicio/Apagado* del icono *Sistema* del *Panel de control* (figura 1).

El Registro sustituye a los archivos .ini de toda la vida

En la sección *operating system* especificaremos en cada entrada un directorio desde el cual se cargará el sistema operativo. Estas entradas admiten las siguientes opciones:

- /Basevideo. Esta opción obliga al sistema a ser configurado en formato estándar de vídeo VGA con una resolución de 640x480. El programa de instalación añade siempre una entrada en el boot.ini con esta opción, que se utiliza para corregir problemas de presentación que pudieran acaecer si el adaptador de vídeo se desconfigura o se configura con valores inadecuados.
- /sos. Esta opción hace que el proceso de arranque muestre los nombres de las unidades que están siendo cargadas (si no figura, únicamente aparecerán unos puntos suspensivos en la parte superior izquierda de la pantalla).

Listado 3. Tipos de datos en el Registro

Tipo de Dato	Descripción
REG_BINARY	Datos de tipo binario. Lo mejor es no tocarlos, y si necesitamos hacerlo, mejor con el panel de control o alguna utilidad que permita modificarlos.
REG_DWORD	Representa un dato de 4 bytes que puede presentarse en formato binario, hexadecimal o decimal.
REG_EXPAND_SZ	Cadenas de caracteres que contienen variables del sistema.
REG_MULTI_SZ	Puede contener varias cadenas de caracteres que contienen una lista de valores.
RES_SZ	Una cadena sencilla de caracteres.

- /noserialmice=[COMn|COMx,y,z]. Esta opción deshabilita la comprobación del puerto de serie indicado. Por ejemplo, /noseriamice=COM2 deshabilita la comprobación de COM2 y /noserialmice=COM2,3 deshabilita la comprobación de los puertos COM2 y COM3. Si no ponemos este parámetro, la comprobación del ratón se deshabilita en todos los puertos de serie. Este parámetro es útil si NTDETECT identifica incorrectamente un puerto de serie para el ratón.

- /crashdebug. Esta opción habilita la capacidad de Reinicio y Recuperación automática.

- /nodebug. Esta opción se utiliza para suprimir la presentación de la información de eliminación de errores. Esta presentación es muy útil para los programadores, pero hace que el sistema operativo vaya ligeramente más lento, así que si no programamos en ese equipo, mejor le ponemos el /nodebug.

- maxmen:n. Esta opción especifica la cantidad de memoria RAM en KB's que utilizará Windows NT.

- scsiordinal:n. Esta opción se utiliza para identificar el adaptador SCSI. Podemos hacer uso de ella si se nos da el caso de tener dos adaptadores SCSI idénticos. La *n* será cero para el primer adaptador.

EL REGISTRO

La base de datos del Registro es el archivo de configuración unificado de Windows NT y también de Windows 95. En él se encuentra el equivalente a los archivos CONFIG.SYS, AUTOEXEC.BAT, WIN.INI y protocol.ini de las versiones anteriores de Windows. Sustituye, en general, a todos los archivos .INI.

Podemos ver el Registro desde el *Editor de Registro de Configuraciones*. Para echarle un vistazo, podemos escribir regedit.exe desde la opción *Ejecutar* del menú *Inicio*.

Regedit.exe se encuentra en el directorio \winnt.

En la figura 2 podemos ver cómo la ventana de esta aplicación está dividida en dos partes. En la parte izquierda podemos ver las categorías de la base de datos del registro mientras que en la derecha podemos ver el valor de las variables de entorno. En el Listado 2 tenéis una descripción de las categorías del Registro.

Las categorías del Registro están jerarquizadas en subcategorías. Por ejemplo, si pulsamos sobre el signo + de HKEY_CURRENT_CONFIG veremos las subcategorías Display, Enum, Software y System.

Cada una de estas subcategorías puede tener a su vez otras nuevas subcategorías en su interior.

Algo que poca gente sabe es que Windows NT dispone, además del *Editor del Registro de Configuraciones* (REGEDIT.EXE), de otro llamado *Editor del Registro* (REGEDT32.EXE), que se encuentra en \winnt\system32\. La diferencia entre ambos es que regedit.exe es más bien un visor con el que ocasionalmente podemos cambiar algún valor y regedt32.exe es realmente el editor del Registro. Como vemos en la figura 3, en este editor las ventanas están superpuestas en cascada y cada una de ellas contiene una de las categorías principales con sus subcategorías desplegadas, lo cual nos puede facilitar bastante el trabajo, sobre todo para no perdernos en la inmensidad del árbol. Al igual que con regedit.exe, a la izquierda tenemos categorías y a la derecha valores.

Para poder cambiar valores en el editor del Registro, abriremos el menú *Opciones* y desmarcaremos el *Modo de sólo lectura*. De este modo el programa se asegura que no se realicen modificaciones involuntarias. A pesar de todo, si metemos la pata al modificar una entrada del Registro y en el siguiente inicio se nos produce algún error de tipo *Critical*, siempre podemos escoger la última configuración buena conocida pulsando la barra espaciadora durante el proceso de arranque. De este modo se perderán los cambios que motivaron el error. Los datos que se presentan en el Registro son ciertamente crípticos para expertos y profanos. Veamos cómo se organiza la información de la parte derecha de la ventana:

- Nombre de la entrada
- Tipo de datos con los que está formada la entrada
- Valor

En la figura 3 podemos ver en HKEY_USERS\DEFAULT\Software\Microstoft\Clock la entrada iFormat es de tipo REG_EXPAND_SZ y su valor actual es 1. ¿Qué quiere decir esto de REG_EXPANZ_SZ? Pues que el tipo de datos es una serie de caracteres que contiene una variable de entorno del sistema. En el listado 3 podéis ver una descripción de los cinco tipos de datos posibles en el Registro.

TOCAR EL REGISTRO

A la hora de tocar el Registro debemos tener en cuenta un par de cosas: la primera es que hay que hacerlo con cuidado, la segunda es que para cambiar las cosas te las sabes o no te las sabes. El Registro pretende ser muy flexible, lo cual nos lleva a que sea complicado de manejar. De hecho ya están apareciendo libros en el mercado dedicados única y exclusivamente al Registro de Windows.

Para muestra, veamos un botón:

Abrimos HKEY_CURRENT_USER, elegimos *ControlPanel* y después *Colors*. En la parte derecha de la ventana veremos una serie de entradas REG_SZ, esto es, cadena sencilla de caracteres.

Si pinchamos sobre la entrada Background:REG_SZ:0 0 0 estaremos modificando el color de fondo del escritorio. Le debemos pasar un valor RGB, que se compone de tres valores que representan la intensidad de los colores rojo, verde y azul. Sus valores van del 0 al 255. En programación se suelen pasar tres valores de tipo byte, esto es, del 0 al 255. Pero en el Registro no; pasamos esos valores como cadena de caracteres. Esto viene a demostrar lo que ya decíamos antes: *el Registro hay que sabérselo.*

Mediante el Registro podemos tocar valores que no son accesibles de otra manera

En este ejemplo, para poner el fondo del escritorio blanco pondríamos Background:REG_SZ:255 255 255.

Para que podáis ir abriendo boca con el Registro, a continuación tenéis un par de cosas de las cuatro que un servidor sabe hacer con el Registro.

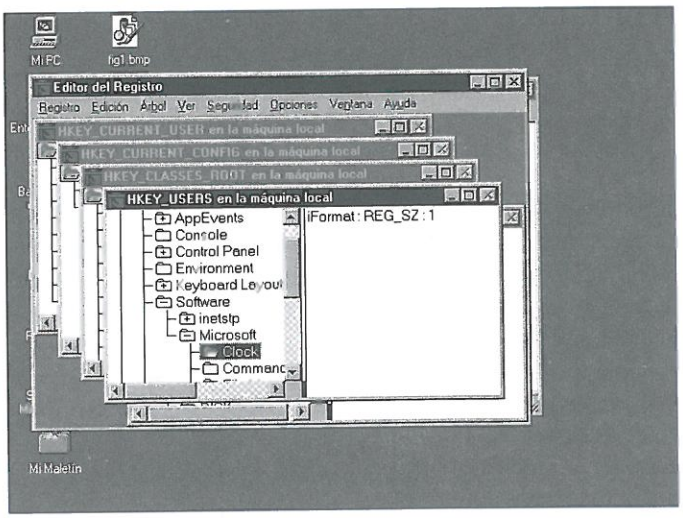


Figura 3.
En el registro nos podemos encontrar con cinco tipos de datos distintos.

MOSTRAR UN MENSAJE DE CONEXIÓN PERSONALIZADO

Podemos mostrar un mensaje personalizado que se mostrará como un cuadro de mensaje al arrancar; después de haber pulsado las teclas (Control)+(alt)+(supr) y antes de mostrar la pantalla de inicio de sesión. Para hacer esto, y personalizar otros parámetros relativos a las conexiones de Windows NT, ejecutaremos regedt32.exe y localizaremos la categoría HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WINDOWS NT\CurrentVersion\winlogon

A continuación hacemos doble clic sobre LegalNoticeCaption:REG_SZ: y escribiremos el texto que aparecerá en la barra de título de la ventana del Mensaje. Por Ejemplo "Inicio de sesión". En la entrada LegalNoticetext:REG_SZ: escribiremos el propio texto del mensaje de conexión (figura 4).

Si nos cansamos de este mensaje no tenemos más que dejar ambas entradas en blanco para que no aparezca esta pantalla.

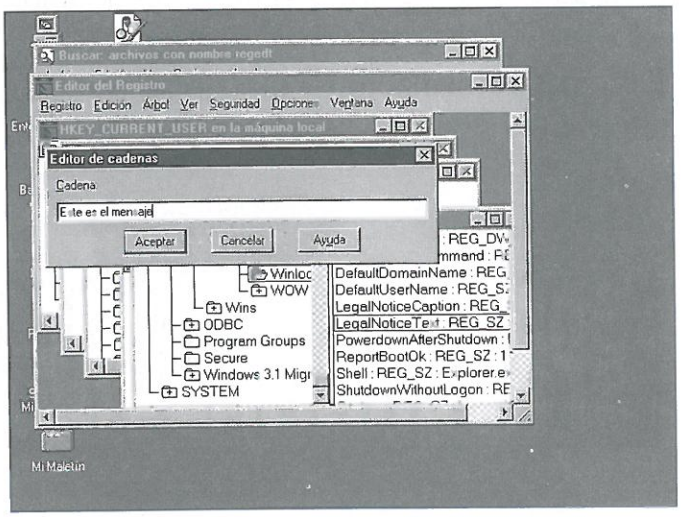


Figura 4.
El editor de cadenas nos sirve para establecer los valores.

REALIZAR UNA CONEXIÓN AUTOMÁTICA

Si la seguridad en el acceso, por lo que sea, es algo que no nos preocupa, podemos hacer que Windows NT inicie sesión automáticamente con un nombre de usuario y una contraseña predeterminados.

Para hacerlo, y sin movernos de donde nos quedamos en el ejemplo anterior, tenemos que seleccionar la entrada DefaultUsername:REG_SZ: y ponemos el nombre de usuario con el que se establecerá inicio de sesión automático.

Luego, si es necesario, otro paso que deberemos dar a continuación, consiste en establecer el valor de DefaultDomainName:REG_SZ: y hacemos lo propio para el dominio al que se conectará el usuario.

Ahora es el momento de crear dos nuevas entradas:

Seleccionamos la opción *Agregar valor* del menú *Edición*. Introducimos el nombre de entrada y el tipo de dato DefaultPassword:REG_SZ y pulsamos Intro Se nos mostrará el editor de cadenas e introducimos el *password* que se utilizará para la conexión automática.

Si cometemos fallos al modificar el Registro, el sistema puede volverse sutilmente inestable

Por el mismo método añadiremos la entrada AutoAdminLogon:REG_SZ:1. En este caso el 1 permite realizar la conexión automática y el 0 lo impide.

Como podéis ver, se pueden hacer muchas cosas con el Registro; incluyendo tocar valores que habitualmente no se permite, porque la única manera de cambiarlos es a través del Registro. Podemos, por ejemplo, cambiar el comportamiento del protocolo PPP, registrar sucesos...

Las pruebas sin saber muy bien lo que se hace están totalmente desaconsejadas con el Registro. En estos casos, lo mejor que nos puede pasar es que nos dé un error de tipo *critical*, porque entonces sabes de seguro que te has equivocado y en qué te has equivocado. Lo malo es cuando ni siquiera se produce un error y lo que ocurre es que el sistema se nos vuelve misteriosamente inestable y empieza a dar los famosos errores de 'pantalla azul' (el sistema se reinicia) sólo cuando confluyen una serie de circunstancias.

En fin, mucho cuidado y mucha copia de seguridad del registro para poder restaurarlo.

Concurso de Programación

Como recordaras teníamos preparado un concurso de programas de ajedrez, pues bien .... el mismo queda convocado para los dias 12 y 13 de Septiembre, siendo la fecha límite de recepción de programas el dia 9 del mismo.

Para más información e inscripciones, puedes ponerte en contacto con nosotros en el teléfono 91 559 82 86 y pregunta por la Srta. SANDRA.

SÓLO PROGRAMADORES

AWS

C/C++

Programación básica en Windows

Jorge Figueroa (erde@arrakis.es)

Desde el momento en el que apareció el sistema operativo Windows, el lenguaje C/C++ ha sido considerado como el lenguaje de programación estándar para esta plataforma, y siempre ligado a los desarrolladores profesionales. Las principales ventajas que ofrece este lenguaje derivan de aspectos como su excepcional velocidad de ejecución, de la generación de ejecutables de tamaño muy compacto, de su gran flexibilidad y sobre todo, de ofrecer casi la misma accesibilidad al hardware del sistema que el propio lenguaje ensamblador.

Hoy en día, casi todos los programadores del mundo escriben sus programas para Windows de 32 bits y por eso, este artículo se va a enfocar en el aprendizaje de la programación C para Windows 95 y NT. Por este motivo, los que deseen practicar el contenido de los artículos tendrán que disponer de un compilador de C/C++ de 32 bits (la gran mayoría de compiladores modernos lo son).

Para empezar, antes de introducir al lector en la propia programación para Windows, que iremos desmenuzando poco a poco, se van a explicar todos los conceptos básicos y elementales, relativos al propio sistema operativo que son necesarios tener presentes. Se abordarán temas relativos a los recursos generales disponibles (tanto visuales y elementales para su funcionamiento, como de programación) así como el funcionamiento interno del propio lenguaje.

COMPONENTES DE UNA VENTANA

Todos los programas de Windows poseen una serie de características comunes, tales como los bordes de las ventanas, la zona de la barra de título, la zona cliente, los cuadros de control, los menús de opciones, etc. Veamos a continuación una lista más detallada de estas características:

- *Barra de título*
  Aparece en casi todas las ventanas y siempre en la zona superior, ocupando una línea de texto. Contiene el título de la aplicación, de forma que podemos saber rápidamente qué aplicación se encuentra en la ventana.

- *Bordes*
  Los bordes se usan para delimitar las ventanas. Estos pueden ser tanto visibles como transparentes, y la mayoría de programas los usan. Gracias a ellos, el usuario puede diferenciar los límites de cada ventana que pueda haber activa y también permiten que se puedan cambiar a voluntad los valores de anchura y altura.

- *Área cliente*
  Se refiere al área de mayor tamaño de una ventana y es donde la propia aplicación interactúa con el usuario.

La programación para MS-DOS ha pasado prácticamente a la historia, y la práctica totalidad de desarrolladores actuales diseñan su programa para Windows. En este artículo se va a iniciar al lector a la programación en C/C++ bajo este sistema operativo, empezando por los conceptos más elementales.

Tabla 1. Tipos de datos y equivalencias de prefijo.

"a":	Átomo
"b":	Bool (entero).
"by":	Byte (carácter sin signo).
"c":	Carácter (ANSI ASCII 8 bits).
"dw":	Dword (unsigned long).
"fn":	Función.
"h":	Handle (entero sin signo).
"hDC"	Handle a un contexto de dispositivo.
"i":	Entero 32 bits.
"id":	Valor id general.
"l"	Long (64 bits).
"lp":	Long Pointer.
"lpsz":	Puntero Long hacia una cadena de terminación Nula.
"n":	Entero corto.
"np":	Puntero near. Short.
"p":	Puntero.
"pfn":	Puntero de función.
"pst":	Puntero a estructura.
"psz":	Puntero a cadena de terminación nula.
"pv":	Puntero a void.
"s":	Cadena.
"sz":	Cadena terminada en Null (0).
"u":	Entero sin signo.
"v":	Tipo Void.
"w":	Word (sin signo de 16 bits).
"x":	Corto (coordenada X).
"y":	Coordenada Y.

- *Botón de cierre*
  Este botón, que aparece como un pequeño recuadro con una cruz, está situado en el extremo superior derecho de las ventanas (dentro de la zona de título) sirve para poder cerrar la forma rápida la aplicación.

- *Botón maximizar, botón minimizar*
  Aparece junto al de cerrar, a su izquierda y sirve para que el usuario pueda maximizar en pantalla el tamaño de la ventana de la aplicación al máximo que permita el modo de pantalla. Por el contrario, el botón de minimizar permite dejar "aparcada" la aplicación en la que lo pulsamos en la barra de tareas, desapareciendo así de pantalla y dejando más espacio libre para otras ventanas de aplicación.

- *Barra de menús*
  Muchas aplicaciones poseen inmediatamente después de la barra de título una barra con los menús desplegables en los que aparecerán todas las opciones de que dispone el usuario en esa aplicación.

- *Barras de desplazamiento*
  Se encargan de mostrar el contenido de ventana, cuando la información que muestra no cabe en la pantalla.

- *Menú del sistema*
  Toda ventana posee en el ángulo superior izquierdo el llamado botón del menú del sistema, el cual da acceso a unas opciones generales de control de la ventana, tales como cerrar aplicación, mover, ajustar el tamaño, etc.

COMPONENTES DE PROGRAMACIÓN

Todos los elementos explicados hasta ahora eran relativos a los componentes visuales de las aplicaciones, ahora se detallarán los relacionados con la programación de una ventana:

- *Mapas de bits*
  Muchas aplicaciones usan mapas de bits (gráficos) para conseguir una mejor apariencia visual de las ventanas y para hacer más intuitiva la información que se muestra en ellas. El uso más típico de mapas de bits es para dibujar iconos, muy utilizados en Windows para indicar eventos, acciones, etc.

- *Pinceles*
  Los pinceles se usan para rellenar espacios cerrados de pantalla con patrones de color. Un pincel debe poseer un tamaño mínimo de 8 por 8 *pixels* y posee tres características: color, tamaño y modelo.

- *Signos de intercalación*
  Se usan para indicar al usuario en qué lugar se "almacenarán" los datos introducidos por el usuario. Por tanto, es el cursor que se utiliza para el manejo de datos en modo texto.

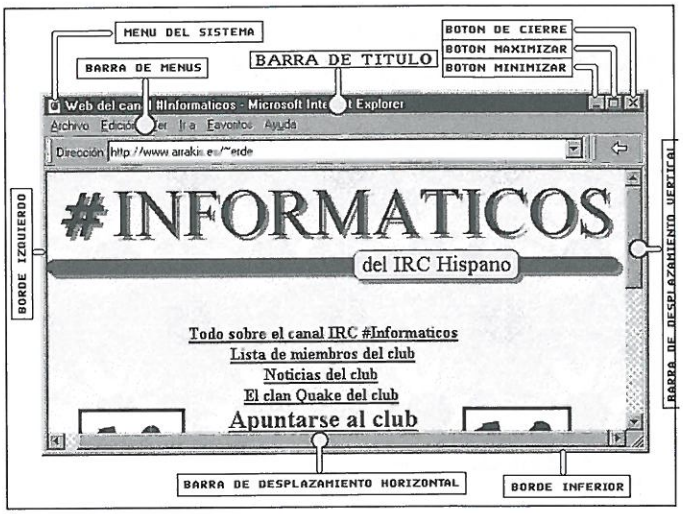
- *Cursores*
  Los cursores son pequeños gráficos que sitúan el puntero del ratón u otro dispositivo similar en pantalla a la vez que cambia de apariencia en función del estado actual del sistema. El modo normal consiste en una flecha (cuando el sistema está ocupado realizando efectuando algún proceso, se sustituye por un reloj).

- *Cuadros de diálogo:*
  Los cuadros de diálogo son pequeñas ventanas usadas básicamente para pedir información (datos) al usuario. Pueden contener muchos elementos como textos, gráficos, cuadros desplegables, zonas de entrada de texto, botones seleccionables, etc., y una característica que los hace fácilmente reconocibles es que siempre poseen botones de salida, como los típicos *Sí*, *No*, *Aceptar*, *Cancelar*.

- *Fuentes*
  Las fuentes de Windows son totalmente programables. En el sistema hay varias definidas por defecto, y se pueden usar para escribir textos con cualquier estilo gráfico, tamaño y atributos (negrita, subrayado, cursiva...).

- *Iconos*
  Los iconos no son más que pequeños gráficos (como ya se ha explicado) que se usan para indicar de forma simple de un evento, situación o idea que la aplicación quiera comunicar.

- *Lápices:*
  Los objetos tipo lápiz se usan para dibujar líneas (ya sean rectas o no). Las funciones de dibujo usan los atributos que tenga el lápiz activo para dibujar. Sus atributos son: color, línea, ancho y estilo de lápiz. El lápiz por defecto usado en Windows es el negro.



Mapa de ejemplo con los principales elementos estándar que aparecen en casi todas las ventanas Windows.

FUNCIONAMIENTO INTERNO DE WINDOWS

La primera gran diferencia entre la programación para DOS y Windows es que en el segundo las aplicaciones no controlan nunca directamente el hardware sobre el que se ejecutan, sino que todo pasa por usar el sistema operativo. Gracias a este tipo de funcionamiento, Windows es capaz de controlar las aplicaciones en todo momento y de poder ejecutar las aplicaciones aprovechando la multitarea.

Todos los programas Windows poseen recursos visuales y de programación comunes

Todos los eventos del sistema: interrupciones, pulsaciones de teclado, movimientos del ratón, etc., son transformados por Windows en lo que se conoce por mensajes. Dichos mensajes se envían a las aplicaciones correspondientes para que éstas los procesen y reaccionen en función del tipo de mensaje recibido. Asimismo, todas las acciones que las aplicaciones realizan, lo hacen por medio de mensajes que mandan al sistema operativo y que éste interpreta para ir efectuar la ejecución de la aplicación.

La primera consecuencia que se deriva de esta forma de funcionar de Windows, es que no es posible ejecutar interrupciones de tipo DOS o BIOS, o lo que es igual, acceder al hardware de forma directa. De esta forma, el sistema intenta evitar la pérdida del control de la ejecución, con el fin de garantiza la protección entre aplicaciones y la estabilidad del sistema.

EJECUCIÓN DE LOS MENSAJES

Como se ha dicho, los mensajes son utilizados por el sistema operativo para informar de los eventos que ocurren a las aplicaciones y para recibirlos de éstas. Por tanto, son el medio utilizado para la comunicación y sincronización de tareas. El funcionamiento interno del manejo de mensajes es muy fácil de comprender. El sistema operativo posee una cola de mensajes y cada aplicación que haya abierta en un momento determinado posee la suya propia (podríamos llamarlas colas locales de aplicación).

Así, tenemos que los mensajes (eventos) que se van produciendo en el sistema se van acumulando en la cola de mensajes del sistema y a medida que el sistema operativo los va procesando, éstos van siendo transferidos a las colas de mensajes de las aplicaciones a las que corresponda. Una vez allí, la propia aplicación se encargará de ir leyendo los mensajes que vayan entrando en su cola local para ir ejecutando las acciones pertinentes.

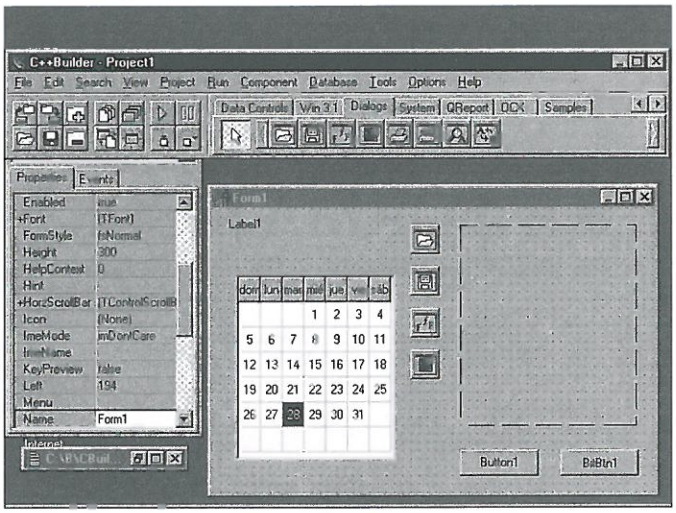
FORMATO DE UN MENSAJE

Todos los mensajes usan un formato común que consta de cuatro parámetros de 32 bits. Se trata de los siguientes:

- Handle: Indica a qué ventana pertenece el mensaje, de forma que así el sistema podrá saber a qué cola local de aplicación tendrá que enviarlo.

- Tipo: El tipo de mensaje será uno de los especificados en la cabecera WINDOWS.H que como se verá más adelante, se ha de incluir en todos los programas C/ C++. El tipo de mensaje se representa mediante dos letras seguidas de un guión bajo del tipo "_" y un texto descriptivo del propio tipo. Ejemplos de mensajes son: WM_PAINT, WM_CLOSE, WM_CUT, etc.

- LPARAM, WPARAM: Estos dos parámetros se utilizan como contenedores de información relacionada con el tipo de mensaje, y que se



Prácticamente todos los compiladores de C/C++ actualmente soportan la programación para Windows de 32 bits.

necesita para llevar a cabo la comunicación de forma correcta. Por ejemplo, podría tenerse un mensaje de pulsación de una tecla, el cual, indicaría el código de la tecla pulsada en LPARAM.

GENERACIÓN Y EJECUCIÓN DE MENSAJES

Windows es un entorno multitarea gracias a que puede controlar todos los mensajes que se generan en el sistema, gestionando el funcionamiento completo del sistema y de las aplicaciones. Los mensajes pueden generarse a partir de cuatro situaciones: que sean generados por el propio usuario, por Windows, por la aplicación activa o por otro programa.

Todos los programas Windows contienen un procedimiento encargado de crearse a sí mismos y de la inicialización. Después de este proceso, poseen una parte de código de programa dedicado al procesamiento de los mensajes, el cual, se encargará de mandar cada mensaje a Windows para que éste lo envíe al procedimiento de ventana correspondiente. Una vez allí, la aplicación se encargará de procesarlo según las características del propio mensaje. A este bloque de programa dedicado al procesamiento de mensajes se le llama bucle de procesamiento de mensajes.

Otra consecuencia de la capacidad de Windows de soportar multitarea, reside en la posibilidad de que el sistema reciba mensajes procedentes de diversas aplicaciones con un orden de llegado que los entremezcle. En este sentido, Windows deberá procesarlos en función de la prioridad de cada uno de ellos, con el fin de reaccionar de forma más rápida con los eventos de mayor trascendencia y prioridad. Debido a esta circunstancia, cada mensaje tiene asociada una prioridad de envío, que puede encontrarse entre estos cuatro valores:

Normal_priority_class.
Idle_priority_class.
High_priority_class.
Realtime_priority_class.

TIPOS DE DATOS USADOS EN WINDOWS

En la programación C/C++ para Windows la definición de variables siempre se realiza siguiendo una norma de codificación: se definen con una más o letras en minúsculas iniciales que describen el propio tipo de dato de la variable y después de esta/s letra/s se indica el propio nombre de la variable. Este sistema de definición se diseñó porque de esta forma hay menos posibilidades de que el programador haga un uso equivocado de una determinada variable, es decir, el uso de un tipo de datos diferente al diseñado en un principio para dicha variable.

En la tabla 1 se muestra la lista completa de tipos de datos usados en Windows y sus prefijos de definición asociados.

EL PRIMER PROGRAMA C PARA WINDOWS

En la tabla 2 se ha incluido el código C correspondiente a un ejemplo de programa básico para Windows en el que aparecen únicamente los elementos esenciales para su ejecución. Una vez ejecutemos el programa, veremos que su utilidad es simplemente para crear una ventana simple (básica). De lo anterior se deduce que este programa se puede usar como esqueleto base para crear cualquier nuevo proyecto que el lector pueda emprender, de forma que pueda practicar la creación de programas completos (que funcionen) antes incluso de que entienda el funcionamiento de esta parte del código inicial.

LA FUNCIÓN WINMAIN( )

Todos los programas diseñados para Windows 95 y Windows NT poseen la función WinMain( ). Esta función define el cuerpo principal del programa, y su funcionamiento y empleo son similares al main( ) que se usaba en los programas C para DOS, donde todo el cuerpo del programa principal se incluía dentro de esta función.

Algunos Ratones son muy curiosos...

DadvinaFence
Seguridad en sus Aplicaciones.

Uno de los riesgos que sufren las aplicaciones informáticas es la posibilidad de su uso por personas no autorizadas.
DadvinaFence incorpora tres componentes:**TFApplicationLock, TFFormLock** y **TFNetMessenger**, proporcionando al programador un sistema de seguridad activo fácil de integrar y totalmente configurable.

- Versiones para Delphi 3.0/2.0, C++ Builder y Visual Basic 4.0/5.0.
- Asignación de permisos y privilegios sobre los objetos exixtentes en las pantallas de las aplicaciones.
- Autentificación de usuarios mediante el uso de identificación y contraseña.
- Detección de intentos fallidos continuados de entrada al sistema avisando al instante a los administradores de seguridad de las aplicaciones.
- Centralización de los accesos (login) en las bases de datos.
- Registro de entradas y salidas.
- Sistema modular fácilmente actualizable en tiempo de ejecución y de diseño.
  etc...

Dadvina Systems.

Sus clientes se lo Agradecerán

Sólo por 31.500* ptas.

San Toribio 6.
28031 Madrid.

Realice su pedido en el teléfono 91 380 32 46. , o en nuestra Web.

* IVA no incluido.

Tabla 2. Listado con un programa básico de ejemplo (EJEMPLO 1).

```
#include <windows.h>

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
Char szProgName[]= "ProgName";

Int WINAPI WinMain(HINSTANCE hInst, HINSTANCE hPreIns, LPSTR.
LpszCmdLine, int nCmdShow)
{
 HWND hWnd;
 MSG lpMsg;
 WNDCLASS wcApp;

 WcApp.lpszClassName  = szProgName;
 WcApp.hInstance      = hInst;
 WcApp.lpfnWndProc    = WndProc;
 WcApp.hCursor        = LoadCursor(NULL, IDC_ARROW);
 WcApp.hIcon          = 0;
 WcApp.lpszMenuName = 0;
 WcApp.hbrBackground = GetStockObject(LTGRAY_BRUSH);
 WcApp.style          = CS_HREDRAW | CS_VREDRAW;
 WcApp.cbClsExtra     = 0;
 WcApp.cbWndExtra     = 0;
 If(!RegisterClass (&wcApp))
   Return 0;

 HWnd= CreateWindow(szProgName, "Ejemplo 1. Un programa de ejemplo", WSO-
VERLAPPEDWINDOW, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
CW_USEDEFAULT, (HWND) NULL, (HMENU) NULL, (HANDLE) hInst, (LPSTR) NULL);

 ShowWindow(hWnd, nCmDShow);
 UpdateWindow(hWnd);
 While (GetMessage(&lpMsg, NULL, 0, 0))
    {
      TranslateMessage(&lpMsg);
      DispatchMessage(&lpMsg);
    }
 return(lpMsg.wParam);
}

LRESULT CALLBACK WndProc(HWND hWnd, UINT messg, WPARAM wParam,
LPARAM lParam)
{
 HDC hdc;
 PAINTSTRUCT ps;

 Switch (messg)
  {
   case WM_PAINT:
     hdc= BeginPaint( hWnd,&ps);

  // Funciones gráficas por debajo de GDI

  //Funciones gráficas por encima del GDI
  ValidateRect(hWnd, NULL);
  EndPaint(hWnd,&ps);
  break;

  case WM_DESTROY:
   PostQuitMessage(0);
   break;

  default:
  return(DefWindowProc(hWnd, messg, wParam, lParam));
 }
}
```

En Windows, la función WinMain( ) se encarga de realizar las tareas siguientes:

1- Crear e inicializar el bucle de procesamiento de mensajes de programa.

2- Realizar las inicializaciones que el sistema requiera para poder ejecutar el programa.

3- Registrar el tipo de clase de la ventana del programa.

4- Finalizar el programa cuando éste concluye enviando un mensaje WM_QUIT a Windows.

Junto con WinMain( ), se pasan cuatro parámetros:

- hInst: Su función consiste en mantener el handle de la instancia para el programa (Se usa como identificador de la aplicación en Windows).
- hPreIns: Contienen siempre el valor NULL.
- LpszCmdLine: Puntero que señala a una cadena con terminación nula, usada para representar los argumentos que se hayan escrito como parámetros al ejecutar el programa si este fue arrancado desde MS-DOS. En caso de ejecutarse directamente desde Windows, este parámetro será un puntero NULL.
- NCmdShow: Contiene un valor entero usado para representar una constante de Windows.

MÁS DETALLES

Hasta ahora, a lo largo de estas páginas, se han explicado las bases de la programación C bajo Windows. En el próximo número de la revista, aparecerá un artículo que profundizará en las explicaciones presentadas, con el fin de abordar temas más concretos.

Creación de controles ActiveX (y IV)

Jordi Agost (agost@eup.udl.es)

A lo largo del último artículo de esta serie abordaremos temas que complementarán los conocimientos adquiridos en las anteriores entregas. De esta forma, conseguiremos aumentar nuestras nociones de formación sobre los controles ActiveX, tratando de cubrir las opciones aún no explicadas hasta el momento en otros artículos.

ESTABLECER UNA PROPIEDAD COMO PREDETERMINADA

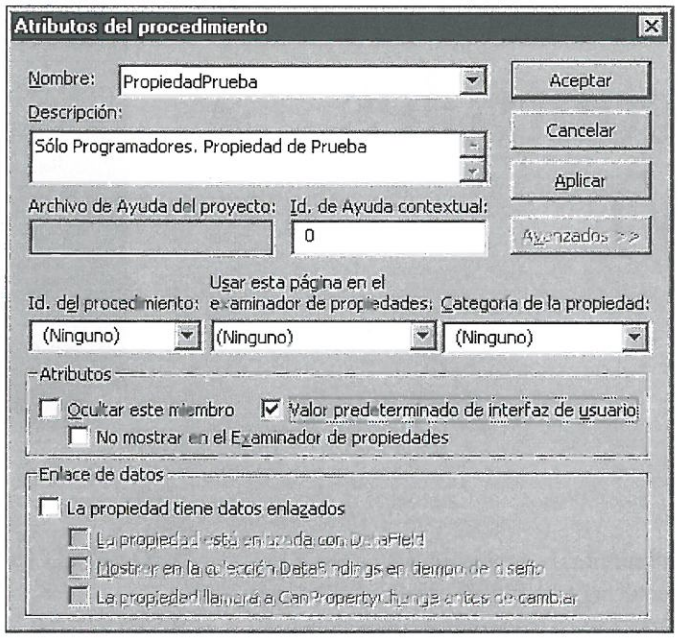
Cuando una instancia de un determinado control se sitúa en un contenedor (usualmente un formulario, aunque no debemos descartar los otros contenedores posibles), Visual Basic elige una propiedad para resaltarla en la ventana de propiedades. Si la propiedad resaltada del último control con el que hayamos trabajado también existe en el nuevo control que acabamos de colocar en el contenedor, entonces será dicha propiedad la que se resaltará. En caso contrario (la propiedad no existe en el nuevo control), Visual Basic elegirá la propiedad que le parezca basándose en criterios internos (propios de VB), tales como el orden en que se agregaron las diferentes propiedades a la biblioteca de tipos.

Los pasos que debemos seguir para realizar dicha tarea son los siguientes: en primer lugar en la ventana de código del objeto **UserControl**, seleccionaremos un procedimiento que corresponda a la propiedad que queremos que aparezca como predeterminada. A continuación en el menú **Herramientas** seleccionaremos el cuadro de diálogo **Atributos del procedimiento**. La propiedad que hemos elegido debería salir en la caja de texto *Nombre,* pero en caso contrario deberemos seleccionarla. Ahora hacemos clic en el botón **Avanzadas** y activamos la casilla **Valor predeterminado de la interfaz de usuario** en el cuadro **Atributos**. Aceptamos y a partir de ese momento la propiedad aparecerá como predeterminada.

Llegados a este punto debemos hacer dos matizaciones: en primer lugar solamente puede haber una propiedad predeterminada de la interfaz de usuario. Si en su lugar queremos añadir otra, entonces la que teníamos como predeterminada dejará de serlo. Y en segundo lugar, comentar que hemos de tener cierto cuidado a la hora de elegir la propiedad predeterminada, siendo la más aconsejada aquella que utilicemos con mayor frecuencia.

AGRUPAR LAS PROPIEDADES EN DIFERENTES CATEGORÍAS

Con el nuevo **IDE** (*Integrated Development Enviroment*: Entorno de Desarrollo Integrado) de Visual Basic 5.0, las páginas de propiedades han pasado a tener ahora



Diálogo Atributos del procedimiento que se utiliza para establecer una propiedad como predeterminada.

dos fichas. La primera es la típica que muestra habitualmente Visual Basic, y en la segunda y nueva, las propiedades aparecen clasificadas por categorías. Podemos asignar la propiedad que creamos a una categoría existente o bien crear nuevas categorías. En el supuesto de que no hiciésemos dicho paso, nuestra propiedad aparecería en la categoría **Miscelánea**.

Debemos cuidar la elección de la propiedad predeterminada

Para que nuestra propiedad esté dentro de una categoría determinada debemos hacer lo siguiente: al igual que en el paso anterior en el objeto **UserControl** (en su ventana de código) colocamos el cursor debajo de la propiedad que queremos clasificar. Seleccionamos el cuadro de diálogo **Atributos del procedimiento** dentro del menú **Herramientas** y nos fijamos en si la propiedad aparece en el cuadro de texto *Nombre*. En caso negativo, la seleccionamos a mano. Acto seguido haremos clic en el botón **Avanzadas** y seleccionamos la propiedad en *Categoría de la propiedad* (ver imagen). Si lo que queremos es crear una nueva categoría, sólo tenemos que escribir el nuevo nombre en la caja de texto de nombres de categoría y automáticamente será creada, aunque debemos tener en cuenta que sólo servirá para este control. Si queremos que la nueva categoría aparezca en un nuevo control, tendremos que volver a escribirla.

Hemos de reflejar las propiedades del nuevo control creado

Un aspecto que se debe tener en cuenta consiste en que somos totalmente libres en el momento de crear nuestro control. En él, podemos incluir las funcionalidades y propiedades que deseemos. No obstante, nuestro sentido común nos dice que hay unas propiedades, que es recomendable incluir. Además, si nuestro control se parece a uno ya existente, es aconsejable incluir las propiedades con que cuenta dicho control. Entre las propiedades de inclusión recomendada destacan: *Appearance* (si la apariencia depende de los controles constituyentes, entonces puede ser problemático incorporar dicha propiedad), BackColor, BackStyle, BorderStyle, Enabled, Font y ForeColor.

CREAR PROPIEDADES SÓLO ACCESIBLES EN TIEMPO DE DISEÑO

Si lo que deseamos es crear una propiedad que pueda leerse en tiempo de ejecución pero que solamente pueda establecerse en tiempo de diseño, emplearemos el procedimiento de propiedades. En el evento Property Let o Property Set (recordemos que si estamos trabajando con propiedades que contengan tipos de datos Variant, tendremos que implementar los dos procedimientos de propiedad: Let y Set, y controlar el error en ambos) comprobaremos la propiedad UserMode del objeto AmbientProperties. Conviene recordar que propiedad UserMode de dicho objeto es False si nos encontramos en fase de diseño y True si nos encontramos en tiempo de ejecución. Si el valor de dicha propiedad es True provocaremos un error. El código podría ser el siguiente:

```
Private VarPrueba As Integer

Public Property Get Prueba() As Integer
   Prueba = VarPrueba
End Property

Public Property Let Prueba(ByVal NewValue As
      Integer)
  If AmbientProperties.UserMode Then
    Err.Raise vbObjectError + 5, "Error. Propiedad
      sólo de lectura en ejecución."
  End If
  Prueba = VarPrueba
  PropertyChanged "Prueba"
End Property
```

En el código de la propiedad Let comprobaremos si estamos en tiempo de ejecución, y si es así, ejecutaremos el método Raise del objeto Err. Recordemos que este método genera un error en tiempo de ejecución (es mejor utilizar Err que la instrucción Error ya que el primero

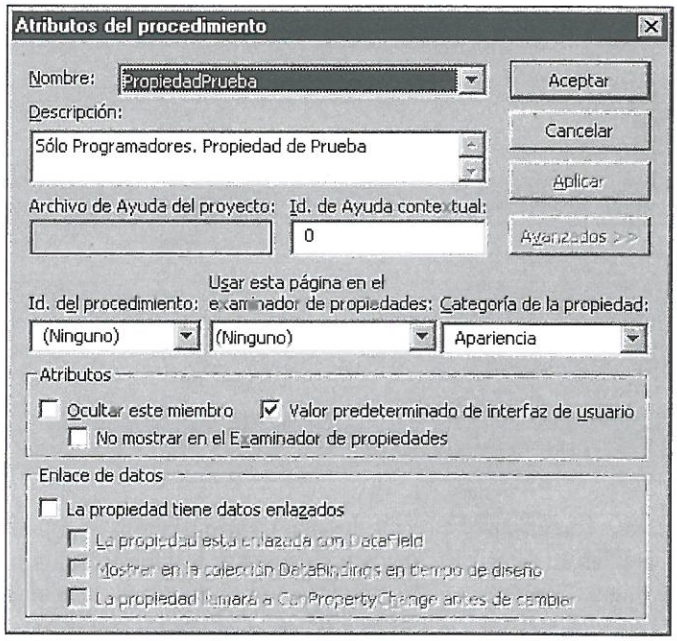
proporciona más información). Su sintaxis es la siguiente:

```
Err.Raise Number [, Source][, Description][,
    Helpfile][, Helpcontext]
```

Donde number será un tipo de datos entero de tipo Long que identificará la naturaleza del error. Los errores de Visual Basic (tanto los definidos por Visual Basic como los definidos por el usuario) se encuentran entre 0 y 65535. Al establecer en la propiedad Number su propio código de error en un módulo de clase, agrega su código de error a la constante vbObjectError. Por ejemplo, para generar el número de error 1050 asignaremos vbObjectError + 1050 a la propiedad Number. Source indica el objeto o la aplicación que generó el error. Description es una cadena que describe el error. Helpfile se refiere a la ruta reconocida del archivo de ayuda de Microsoft Windows en el que se encuentra la información sobre el error. Por último, Helpcontext es el identificador de contexto que especifica el tema del archivo indicado en HelpFile, que a su vez contiene la información de ayuda del error.

PROPIEDADES DE SÓLO LECTURA EN TIEMPO DE EJECUCIÓN

Podemos crear propiedades que sólo se encuentren disponibles en tiempo de ejecución si dentro de los diferentes procedimientos de propiedad controlamos si estamos en fase de diseño o no (será cuando la propiedad UserMode del objeto AmbientProperties sea False). Y además debemos abrir el cuadro de diálogo Atributos del procedimiento seleccionando dentro del menú Herramientas la opción Atributos del Procedimiento. A continuación, elegir la propiedad que queremos que sea visible sólo en tiempo de ejecución, desplegar las opciones Avanzadas y activar la casilla No mostrar en el examinador de propiedades para que la propiedad ya no aparezca en la ventana de propiedades.



Diálogo Atributos del procedimiento, donde se podrá establecer la categoría a la que pertenece la propiedad.

TRATAMIENTO DE LAS PROPIEDADES ESTÁNDAR DE LOS CONTROLES

Si las propiedades que deseamos que posea nuestro control son estándar (como por ejemplo el establecimiento de un determinado tipo de fuente o el establecimiento de un determinado color) colocaremos siempre que sea posible las páginas estándar de propiedades que nos proporciona el propio sistema. Por ejemplo, si añadimos una propiedad llamada MousePointer, tal nos controle la apariencia física del puntero del ratón cuando se encuentra encima del control, y a la propiedad creada le añadimos el siguiente código descrito a continuación:

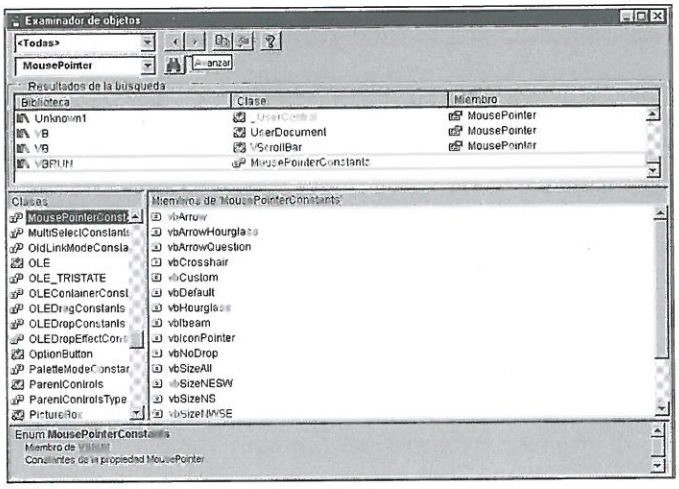
```
Public Property Get MousePointer() As
        MousePointerConstants
  MousePointer = UserControl.MousePointer
End Property

Public Property Let MousePointer(ByVal NewPt
        As MousePointerConstants)
  UserControl.MousePointer = NewPt
  PropertyChanged "MousePointer"
End Property
```

Entonces veremos que cuando dicha propiedad aparezca en la página de propiedades tendrá los mismos elementos que las propiedades MousePointer de otros controles. En el examinador de objetos podemos ver todas las enumeraciones disponibles para Visual Basic.

EL ENFOQUE EN LOS CONTROLES ACTIVEX

Si recordamos los artículos anteriores, veremos que existían tres métodos para crear los controles ActiveX: los controles propiamente dibujados, los controles que



Aspecto del examinador de objetos, con las enumeraciones disponibles para Visual Basic. En concreto vemos las enumeraciones para el ejemplo de MousePointer citado.

a su vez emplean controles constituyentes y los controles que mejoran un determinado control constituyente. En el primer caso inicialmente el objeto UserControl no contendrá ningún control constituyente, por lo que seguramente no nos interesará que dicho control pueda llegar a recibir el enfoque. Para que esto no suceda debemos establecer la propiedad CanGetFocus del objeto UserControl a False. Y además no podemos obviar este paso porque por defecto la propiedad tiene el valor opuesto. En el caso en que estableciéramos la propiedad CanGetFocus a su valor True, el objeto UserControl recibirá los eventos EnterFocus, Gotfocus, LostFocus y ExitFocus cuando el control reciba y pierda el enfoque. EnterFocus se produce antes de que el control tenga el foco, es decir, de que se haya producido el evento GotFocus De igual forma ExitFocus se producirá después del evento LostFocus, más concretamente cuando el hilo del programa vuelva del final de la rutina LostFocus.

Cuando el foco es captado por algún control constituyente del objeto UserControl, éste recibe el evento EnterFocus, y posteriormente el evento GotFocus. Si ahora el foco no se mueve de dentro del objeto UserControl, tan solo se producirán los eventos relacionados con el foco de los controles constituyentes. Para tratar el tema del enfoque en el control no es en absoluto necesario escribir código en dichos eventos (EnterFocus y ExitFocus), es más, los creadores de Visual Basic desaconsejan hacerlo (aunque puede hacerse). De igual forma el utilizar MsgBox al realizar la depuración de nuestro programa, y en el caso en que estemos tratando temas relacionados con el foco, nos puede conducir a errores de interpretación de la depuración, ya que cuando escribimos un MsgBox, la caja que se crea es la que coge el foco, interfiriendo en nuestra depuración. Es mucho más aconsejable utilizar el método Debug.Print.

EnterFocus se produce antes de que el control tenga el foco, por lo tanto, antes de GotFocus

En los métodos segundo y tercero, es decir, si nuestro control utiliza controles constituyentes, o en su defecto mejora un control constituyente ya existente, entonces nuestro objeto UserControl por sí solo no podrá recibir el enfoque (sea cual sea el valor de la propiedad CanSetFocus), a no ser que alguno de los controles constituyentes pueda llegar a recibir el enfoque. Si ningún control constituyente puede recibir el foco y el valor de CanSetFocus es True, entonces el objeto UserControl recibirá los mismos eventos que un control dibujado por el usuario. Y además, si CanSetFocus es False y el objeto UserControl posee algún control tal que pueda recibir el foco, entonces se producirá un error.

CÓMO ESTABLECER TECLAS DE ACCESO AL CONTROL

Podemos establecer teclas de acceso al control (como sucedería por ejemplo en el control CommandButton al colocar un carácter & en su propiedad Caption) modificando la propiedad AccessKeys del objeto UserControl. Pero para tal efecto debemos distinguir dos casos, en primer lugar el caso en el que un usuario crea un propio control y en segundo lugar, el usuario de nuestro control creará un control ActiveX a partir de controles constituyentes.

En el primer caso, supondremos que el usuario ha creado un determinado control, dibujado por él mismo, entonces en el procedimiento PropertyLet de la propiedad Caption podríamos mirar si existe un signo & delante de una letra. En este supuesto le asignaremos dicha letra a la propiedad AccessKey, y si el usuario final oprime dicha tecla entonces el objeto UserControl recibirá el evento AccessKeyPress.

En el segundo caso, es decir cuando nuestro control esté compuesto por más de un control constituyente entonces dichos controles constituyentes, como es lógico, van a recibir el enfoque de forma individual.

Estableceremos teclas de acceso al control modificando AccessKeys

Si creamos para este caso un control con una etiqueta y una caja de texto (llamémosle a la etiqueta lblLabel1 y a la caja de texto txtText1), y queremos que el usuario pueda establecer la tecla de acce-

LA PRIMERA REVISTA DE PROGRAMACIÓN EN CASTELLANO

SOLO PROGRAMADORES

ESPECIAL MONOGRÁFICO Nº 3

JAVA EL LENGUAJE DEL FUTURO

Contiene CD-ROM que incluye:

- ✔ JDK v1.1.6 & v1.2Beta3 & Documentación
- ✔ VisualAge for Java
- ✔ Beans Development Kit (Mar98)
- ✔ Free Builder
- ✔ CORBAplus, Java edition
- ✔ Tutorial sobre Java
- ✔ Y mucho más

A la venta a partir del 15 de julio

por sólo 1.250 ptas. IVA incluido

so al control variando tan solo el título de la etiqueta, entonces tendríamos que crear una propiedad Caption con el código que se mostrará a continuación. Antes de mostrarlo conviene aclarar que esto será posible siempre y cuando el título de la etiqueta contenga el carácter &, de forma que una vez establecida el título de la etiqueta (supongamos que es la propiedad Caption de nuestro control), el foco se traslade a la caja de texto.

```
Public Property Get Caption() As String
  Caption = lblLabel1.Caption
End Property

Public Property Let Caption(ByVal NewCaption As
        String)
  lblLabel1.Caption = NewCaption
  PropertyChanged "Caption"
End Property
```

Y además deberemos modificar las propiedades TabIndex de nuestro control, de forma que la etiqueta tenga la propiedad TabIndex con una valor igual a 0 y la caja de texto TabIndex con un valor igual a 1. (dichas propiedades serán invisibles fuera del control). Con el código anterior, colocaremos el control ActiveX en un formulario, añadiremos un botón de nombre cmdCommand1 en el que ejecutaremos el siguiente código:

```
Private Sub cmdCommand1_Click()
  UserControl11.Caption = "&Prueba"
End Sub
```

Veremos que nuestro control aparece con el texto siguiente: Prueba, de tal forma que al pulsar la combinación de teclas Alt+P el foco se traslada a la caja de texto.

DIBUJANDO UN CONTROL ACTIVEX

Como ya viene siendo habitual cuando dibujamos el control nos encontramos con los dos conocidos tipos de control: por una parte los controles totalmente dibujados por el usuario, en los que éste deberá encargarse de todo: dibujar el control, saber si este está seleccionado o no, dibujar el marco de selección, etc. Por otra parte, tenemos el control ActiveX formado a partir de controles ya existentes en el que tan solo deberemos ocuparnos de que estos se posicionen de forma correcta dentro del objeto UserControl.

Con CommandButton podemos incrementar el tamaño de un pixel

En el primer caso, el de un control dibujado por el usuario, solamente tendremos que incluir código en el evento UserControl_Paint, y el código que hará que el control se dibuje. Además el código tan solo puede ser de dibujo (independientemente de las órdenes que utilicemos para dibujar), evitando a toda costa, si no queremos que se produzcan errores, las llamadas a órdenes tales como Do Events o la llamada a otros procesos. El evento Paint será invocado automáticamente cada vez que el formulario (o en su defecto el contenedor) vuelva a repintar el área donde se encuentre nuestro control ActiveX. Puede darse el caso, en que por una orden interna, el control cambie de estado y nos interese volver a dibujarlo en su nuevo estado. En este caso simplemente tenemos que invocar el método Refresh del objeto UserControl.

En el segundo caso, si lo que queremos es mejorar un control que ya existe, lo habitual será que dicho control ocupe por completo toda la extensión del objeto Usercontrol. Para ello usaremos el método Move del control constituyente en el evento Resize del objeto UserControl, tal y como se muestra en el siguiente ejemplo:

```
Private Sub UserControl_Resize()
  Pic1.Move 0, 0, ScaleWidth, ScaleHeight
End Property
```

En el ejemplo anterior suponemos que nuestro control, es o está contenido en un objeto Picture. Si en lugar de tener un solo objeto tenemos diversos, entonces nos encontraremos con que el procedimiento Resize del objeto UserControl va a ser más complicado, ya que deberemos tener en cuenta varios factores añadidos, como por ejemplo, si existen controles que no se redimensionan, la forma en que se distribuyen entre ellos, el cambio de tamaño de todos ellos, etc. Además deberemos tener en cuenta como "crece" el control. Por ejemplo en un control CommandButton podemos hacer un incremento de su tamaño de un pixel (realmente lo mediríamos con twips) , pero hay controles como por ejemplo un ComboBox o un TextBox que no se pueden redimensionar al tamaño justo que nosotros queremos, sino que tienen unos ciertos incrementos establecidos que no podemos modificar cuando cambiamos su dimensión. Cabe tener en cuenta que las propiedades Width y Height del objeto UserControl siempre se medirán en twips, a no ser que en ScaleMode hayamos definido una unidad diferente, con lo que tendríamos que usar los métodos ScaleX y ScaleY para pasar nuestros cálculos de tamaño a twips.

Width y Height incluyen el grosor del borde suponiendo que éste exista

Otro aspecto a menudo olvidado y que puede llevarnos más de un quebradero de cabeza se refiere a la utilización de la propiedad BorderStyle. Consiste en que las propiedades Width y Height del objeto UserControl incluyen el grosor del borde suponiendo que éste exista. Además si éste es del estilo simple y fijo, es decir, que tenga su propiedad BorderStyle con un valor igual a 1, el área disponible para el depósito de nuestros controles se reducirá en dos pixels (no en dos twips), en alto y en ancho.

Siguiendo con el tema del tamaño de los controles, a veces puede ser interesante conseguir que el control no pueda pasar de un tamaño "mínimo". Esto puede suceder por ejemplo si nuestro control ActiveX se encuentra formado por controles constituyentes, ya que dichos controles podrían dejar de verse. Para evitar este paso deberíamos incluir el siguiente código:

```
Private Sub UserControl_Resize()
  Dim MinH As Integer
  Dim MinV As Integer

  ' MinH = cálculo del mínimo horizontal
  ' MinV = cálculo del mínimo vertical

  If Width < MinH Then
    Width = MinH
  Endif
  If Height < MinV Then
    Height = MinV
  Endif
End Sub
```

Paint aparecerá al repintar el área de nuestro control ActiveX

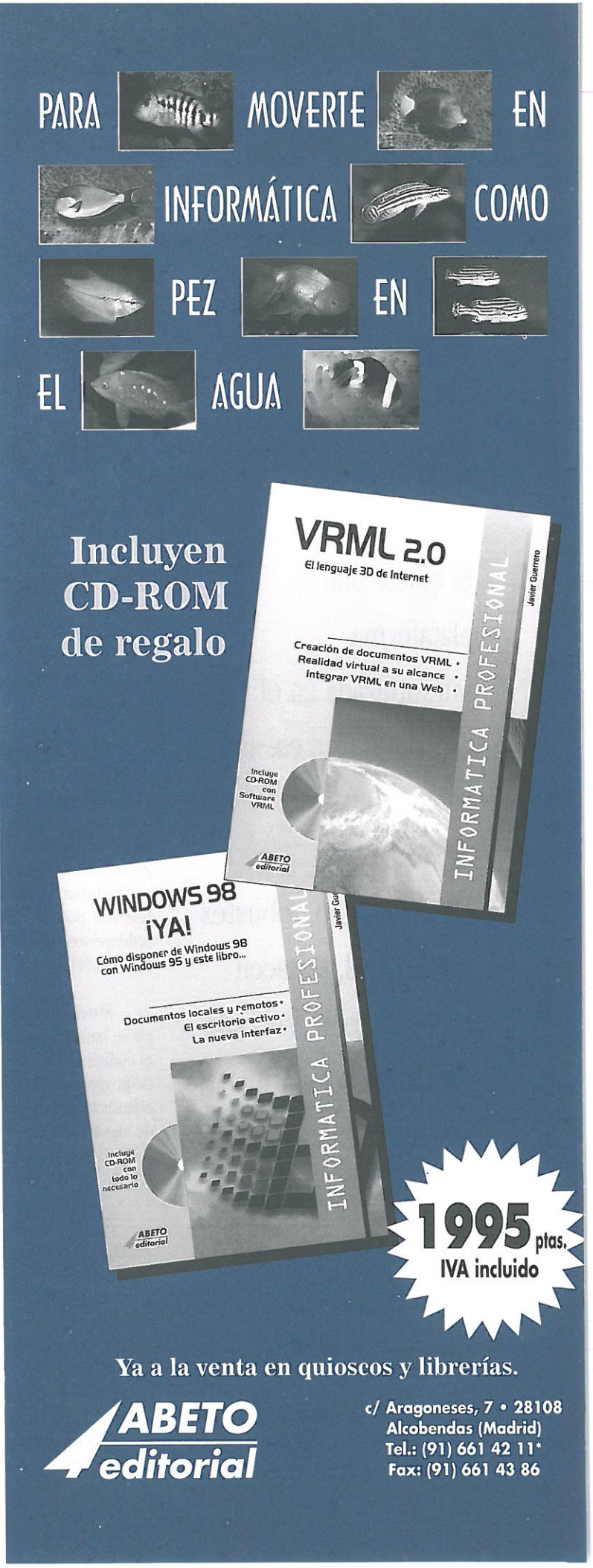
Debemos tener en cuenta dos aspectos: en primer lugar, cualquier llamada al evento Resize del objeto UserControl debe ir acompañada de un cierto control de errores, ya que el contenedor no va a encargarse de dicho control. Si obviamos este paso puede suceder que la aplicación se bloquee. Y en segundo lugar, debemos tener cuidado cuando estemos depurando el control, por ejemplo, en el código anterior dentro del fragmento:

```
If Width < MinH Then
   Width = MinH
Endif
```

Automáticamente se producirá un redimensionamiento del objeto UserControl, lo cual producirá otro evento Resize, hecho que a veces puede llevarnos a la confusión al no saber el trozo de código que estamos depurando.

RESUMEN FINAL

Hasta aquí hemos visto los principales rasgos de la generación de controles ActiveX, con la información dada hasta el momento, ya podemos crear nuestros propios controles para nuestro uso, o para la explotación a terceros.



LINUX

SVGALIB (I): una joya por descubrir

Jorge F. Delgado Mendoza (ernar@convex.es)

Si existe una plataforma infrautilizada en el aspecto lúdico es, sin duda, Linux. Su capacidad de interconexión, robustez y velocidad, la hacen especialmente indicada para la creación de juegos multiusuario a través de Internet. Con una librería gráfica como SVGALIB podremos sacar mucho más provecho a este novedoso sistema.

Con ya casi siete millones de usuarios registrados en todo el mundo, **Linux** se está empezando a convertir en una alternativa seria a los sistemas operativos tradicionales, fundamentalmente gracias a **Internet**.

La unión **Linux/Apache** se ha convertido en el servidor *Web* más extendido; día a día más empresas confían sus servicios de impresión a redes de PC's con máquinas **Linux**, combinando así lo mejor de dos mundos: la robustez y eficacia del Sistema Operativo **Unix** con la gran relación potencia/precio que ofrecen los ordenadores personales basados en la plataforma Intel.

Muchas empresas, incluida **Netscape** se han dado cuenta de las ventajas que el código libre - que no gratuito - es capaz de proporcionar. Poco a poco se hace más común ver PC's con **Linux** en las mesas de desarrolladores que hace cinco años utilizaban estaciones de trabajo de Sun o HP.

Sin embargo, a pesar de todas sus ventajas, existen dos puntos en los que **Linux** se encuentra aún muy lejos de las prestaciones de los famosos Windows. El primero de ellos, probablemente el más importante, consiste en la falta de un conjunto de aplicaciones ofimáticas estándar, semejantes a Office de Microsoft. Este simple hecho hace imposible que el mayor *boom* en el campo de los ordenadores de los últimos tiempos haga su entrada definitiva en el mundo de la productividad empresarial.

El segundo, quizás no tan importante ya que el público objetivo controla una parte mucho más pequeña del pastel, se refiere al apartado lúdico. ¿Qué juegos de calidad podemos encontrar orientados a la plataforma Linux? La respuesta es muy fácil, y no es la que nos gustaría escuchar: pocos.

En la actualidad, siete millones de usarios trabajan con el sistema Linux

Podemos encontrar versiones de los archiconocidos *Doom I*, *Doom II* y *Quake*. Existen además clones de *Civilization* y de *Sim-City*, así como una versión del famoso emulador de máquinas de videojuegos Mame. Algunos otros juegos han sido también portados a *Linux*. Entre ellos destacan, por su cuidado nivel de detalle, *Maelstrom* y *Abuse*.

¿Cómo es posible?, es la primera pregunta que nos podríamos hacer, ya que Linux tiene la reputación de ser más rápido y estable que la plataforma de **Microsoft**. Posee una arquitectura de comunicaciones integrada en el sistema operati-

vo que facilita enormemente la creación de juegos cooperativos. El sistema de ventanas que ofrece, *X-Windows*, permite lanzar ventanas a través de una red de ordenadores simplificando la transmisión de información entre los diferentes jugadores. Además, todo el código es de libre distribución, gracias a lo cual el tiempo de desarrollo se podría reducir de manera considerable.

Una respuesta que explica la falta de éxito de Linux como plataforma lúdica podría encontrarse en el hecho de que Linux, hasta ahora, nunca ha sido un negocio. Las casas comerciales jamás han considerado este sistema operativo como una fuente de ingresos viable, aunque este aspecto podría cambiar a la vista de la expansión de Linux.

Se caracteriza por ser más rapido y estable que la plataforma Microsoft

Si queremos dar una respuesta más detallada, deberemos realizar un análisis más profundo de las circunstancias que rodean este sistema operativo único.

- *Libre Distribución*: es el concepto fundamental en el que se basa, sin el cual nunca habría llegado a ser lo que es hoy en día. Sin embargo, esta circunstancia también influye de forma negativa, ya que las casas comerciales desconfían de un entorno en el cual no existe un soporte técnico claro y definido. Por otro lado, los usuarios de Linux no están acostumbrados a pagar por el software que utilizan, por lo tanto son reacios a utilizar paquetes comerciales.

- *Arquitectura*: concebido mayoritariamente como un entorno de desarrollo, las facilidades que ofrece el sistema operativo a los creadores de juegos son escasas. La consola sólo permite utilizar el modo de texto y el entorno de ventanas. Aunque más avanzado y capaz que el de Windows, no está diseñado para permitir accesos directos a la memoria de la tarjeta, sino que ofrece una serie de funciones primitivas básicas de dibujo a todas luces insuficientes para la creación de un juego moderno.

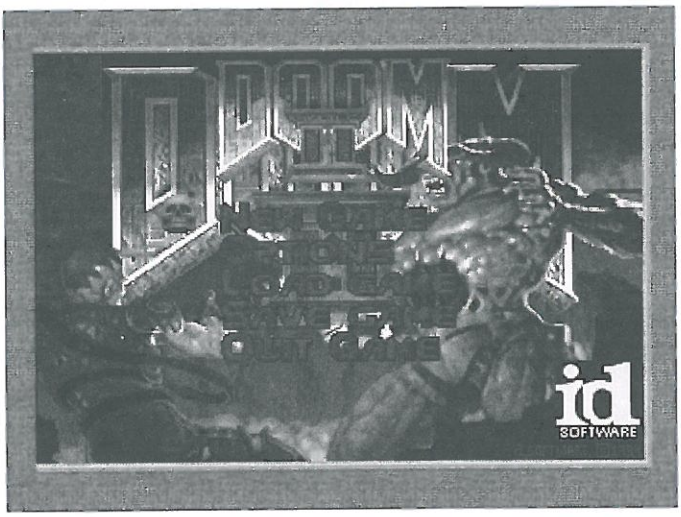


Figura 1.
Los juegos Doom y Doom II fueron pioneros en su género. Además de ello, fuero los primeros juegos comerciales de gran éxito en ser portados a Linux a los pocos meses de su introducción en el mercado Wintel[1].

- Hasta ahora, Linux ha sido un sistema operativo de *nicho*, con un número muy reducido de usuarios. Este motivo es, por sí solo, más que suficiente para que cualquier empresa que se planteara la creación de una versión de sus programas para Linux, se pensara dos veces su viabilidad económica.

- El número de personas con experiencia en el sistema operativo era pequeño, por lo que las compañías se encontraban con la necesidad de contratar gente capaz de llevar el desarrollo en este nuevo entorno. Dado el origen universitario de Linux, encontrar profesionales que cumplieran todos los requisitos exigibles era una tarea complicada.

- Finalmente, el número de aplicaciones de soporte a la hora de construir un juego ha sido siempre muy pequeño. Ninguna casa de software se puede plantear el desarrollo de las herramientas de diseño además del juego en sí. Simplemente no es rentable.

Sin embargo, las circunstancias que hemos descrito están cambiando a una velocidad realmente asombrosa. Por un lado, el número de usuarios de Linux ha aumentado exponencialmente. En cinco años ha pasado de unos pocos usuarios a varios millones. Ni siquiera Windows NT, con todo el empuje económico y publicitario de Microsoft detrás, ha sido capaz de extenderse de esa manera.

La libre distribución se ha convertido en uno de los conceptos fundamentales para entender Linux

Por otro lado, el perfil del usuario de Linux ha cambiado. Ya no es el de un universitario con poca capacidad adquisitiva, sino que cada día es utilizado por un mayor número de profesionales, capaces de valorar las ventajas de adquirir software comercial desarrollado expresamente para la plataforma. Además, con el paso de los años, un gran número de universitarios ha finalizado sus estudios, creando un frente de influencia pro-Linux en sus em-

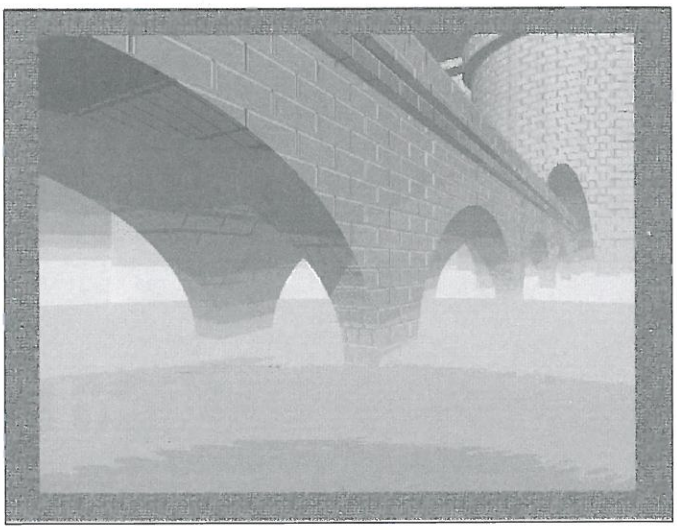


Figura 2. Hoy en día existen una gran cantidad de herramientas gráficas para Linux. Como ejemplo tenemos esta imagen construida mediante el procedimiento de traza de rayos de luz o raytracing, utilizando el programa Persistence Of Vision.

presas. Como efecto secundario, el número de personas con experiencia profesional en el entorno crece día a día.

Adicionalmente, a lo largo de los cinco años de vida real del sistema operativo, han aparecido diversas herramientas -tanto comerciales como gratuitas- capaces de satisfacer las necesidades más exigentes. En el caso que nos ocupa hoy en día existen Raytracers[2] de calidad profesional, aplicaciones CAD, entornos IDE[3] capaces de rivalizar con sus homólogos de Windows, así como herramientas RAD[4] de alta calidad.

Solo quedan, pues, dos escollos que salvar para convertir Linux en una plataforma lúdica viable. El primero de ellos es atraer la mirada de las compañías que los diseñan, siguiendo el ejemplo de la compañía Id Software, creadora de juegos de la categoría de Doom y Quake.

La posibilidades para crear juegos en sistema Linux son más bien escasas

El segundo es la falta de una arquitectura diseñada para soportar este tipo de aplicaciones. Desde cierto punto de vista Linux se enfrenta al mismo tipo de problemas que MicroSoft se encontró con Windows 95 al poco tiempo de su comercialización. Por un lado, la GUI de Windows 95 era demasiado lenta para las exigencias de los juegos modernos, lo que motivaba que la mayoría de ellos se continuaran realizando bajo MS-DOS. Por otro, los juegos de MS-DOS exigían resoluciones más allá del estándar VGA, el cual se limita a definir modos de hasta 320x200 pixels y 256 colores.

Las soluciones que se adoptaron fueron dos: la Video Electronics Standard Association definió una API[5] de acceso a diferentes modos de vídeo avanzados, naciendo así los famosos drivers VESA. MicroSoft se dio cuenta de que si quería que Windows 95 se impusiera en todos los campos relacionados con los ordenadores PC, debía facilitar la creación de juegos bajo su plataforma. Nacieron así DirectX y Direct3D, las cuales permiten a los programas utilizar directamente la tarjeta de vídeo, - como es el caso de DirectX, - o bien acceder a un coprocesador 3D en el caso de tenerlo instalado, - mediante Direct3D. De esta forma los juegos, y otras aplicaciones críticas, pueden acceder directamente al subsistema de vídeo del PC sin que les afecte la sobrecarga añadida del entorno gráfico.

En el caso de Linux la situación ha seguido un proceso similar. A pesar de la aparición de algunas herramientas orientadas a una eficiente presentación de gráficos, como pudieran ser las extensiones de memoria compartida (SHM), la sobrecarga que introducen las primitivas del servidor de X-Windows son demasiado elevadas.

De una manera similar a las DirectX de Windows 95, apareció en XFree86 la extensión DGA, (Direct Graphic Access). Sin embargo, debido a su reciente aparición, aun no es posible utilizarla para productos de calidad comercial.

Además de DGA, el boom de las 3D en el mundo de los ordenadores ha impulsado a numerosos programadores a portar la famosa librería gráfica de Silicon Graphics, OpenGL, a Linux. La versión de distribución gratuita se denomina Mesa.

Aunque pudiera parecer que Mesa iba a ser la solución a los problemas planteados, la falta de información libre sobre las tarjetas aceleradoras 3D, hace muy difícil conseguir un rendimiento adecuado para los juegos de reciente aparición.

En realidad, lo que Linux necesita es una librería que permita acceder directamente a la tarjeta gráfica, sin la sobrecarga del entorno de ventanas, pero que la vez permitiera abstraernos del hardware que tenemos por debajo. Si además la librería utilizara el coprocesador gráfico de la tarjeta de vídeo en caso de existir, utilizando rutinas software para aquellas funciones que no realice la tarjeta, tendríamos la herramienta perfecta para nuestros propósitos.

Con una arquitectura adecuada, se lograría una plataforma lúdica con muchas posibilidades

Bueno, pues aunque parezca mentira, una herramienta como la anterior existe y se llama SVGALIB. Como su propio nombre indica, es una librería para tarje-

tas SVGA, que permite utilizarlas de una manera rápida y eficiente, sin necesidad de conocer cómo son por dentro. En algunos casos, la librería utiliza funciones de los aceleradores de las diferentes tarjetas. A lo largo de los próximos artículos, vamos a aprender a utilizar la librería gráfica para crear nuestras propias aplicaciones. Comenzaremos introduciendo los aspectos más básicos como pueden ser la inicialización de la librería, la selección de los modos de vídeo, etc. En los próximos artículos abordaremos temas más avanzados, incluyendo las primitivas de dibujo, el manejo de los diferentes periféricos, etc.

REQUISITOS

- Conocimientos: la librería gráfica SVGALIB ha sido desarrollada, como el resto del sistema operativo Linux, utilizando el lenguaje de programación C. Los motivos de esta elección son principalmente dos:
  - Por un lado, la baja sobrecarga de C permite extraer el máximo jugo de los diferentes algoritmos de dibujo, sin que se pierda demasiado rendimiento con respecto a la utilización directa de lenguaje ensamblador. En una librería de bajo nivel como ésta, es imprescindible conseguir la mayor velocidad posible aunque ello perjudique la comodidad en el desarrollo.
  - Por otro lado, el compartir entorno de desarrollo con el sistema operativo, permite acceder de una manera cómoda a un gran número de primitivas de bajo nivel, incluyendo aquellas que manipulan directamente los diferentes registros de las tarjetas.

  Además, es muy interesante estar familiarizado con las herramientas de desarrollo de Linux: el compilador de C y la herramienta Make.

  Un conocimiento básico de cómo funciona el sistema de señales Unix, así como de las funciones asociadas a él, sería muy interesante. Entre las llamadas con las que conviene estar familiarizado se incluyen:
  - signal()
  - select()
  - sigaction()

  Finalmente, es muy importante conocer el subsistema de vídeo de un PC. Tener un conocimiento detallado de los diferentes modos que hay, sus distintas resoluciones y colores, así como la cantidad de memoria que debemos tener instalada para cada uno de ellos, es imprescindible para diseñar un sistema gráfico atractivo.

Necesita una librería que permita acceder directamente a la tarjeta gráfica, sin sobrecarga del entorno

- Software: necesitaremos tener instaladas las siguientes herramientas:
  - gcc 2.7.0 o superior.
  - make.
  - SVGALIB 1.2.10 o superior.
- Documentación: prácticamente no va a ser necesario ningún tipo de documentación más allá de los artículos de la revista. A lo largo de ellos vamos a intentar exponer la metodología de diseño gráfico de una manera gradual, apoyándonos en los conocimientos ya adquiridos a la hora de profundizar en la materia. Sin embargo, si alguien está muy interesado en ir más allá de lo que vamos a profundizar en la revista, puede sentirse libre de contactar conmigo mediante correo electrónico, utilizando la dirección indicada.

  Como ya hemos comentado anteriormente, es muy importante conocer, aunque solo sea superficialmente, la arquitectura de vídeo de un PC. Este conocimiento nos permitirá diseñar nuestra aplicación de una manera óptima. Es importante hacer notar que existen pocos programas que machaquen un ordenador más que un juego de última generación. Existen numerosos libros que tratan este tema con diferente grado de profundidad, así como gran cantidad de material accesible en Internet. Probablemente una de las mejores direcciones que se pueden encontrar sobre el tema es:

  http://www.cs.umu.se/~christer/GR/development.html

  En ella podremos encontrar una gran cantidad de información técnica, así como enlaces a un gran número de páginas relacionadas con el mundo de los gráficos.

SVGALIB: QUÉ ES Y PARA QUÉ SIRVE

En la introducción del artículo se comentó que SVGALIB es una librería gráfica de bajo nivel para tarjetas SVGA que trabajan bajo plataformas Linux/Intel. Su misión principal consiste en proporcionar una interfaz común de programación, ocultando al usuario las diferencias existentes entre las diversas marcas y modelos de tarjetas de vídeo disponibles en el mercado.

SVGALIB es una librería gráfica ideal para trabajar con Linux

Tabla 1. Listado de los componentes esenciales de SVGALIB. En cada uno de los casos se indica el tipo de fichero y su ubicación final en el árbol de directorios de la máquina.

Fichero	Tipo	Directorio
vga.h	cabecera	/usr/local/include
vgagl.h	cabecera	/usr/local/include
vgakeyboard.h	cabecera	/usr/local/include
vgamouse.h	cabecera	/usr/local/include
libvga.so.*	dll	/lib
libvgagl.so.*	dll	/lib
libvga.a	librería	/usr/local/lib
libvgagl.a	librería	/usr/local/lib
libvga.config	configuración	/etc/vga

Gracias a ella, la persona que desarrolla la aplicación - o el juego - no tiene que preocuparse de las tareas de inicialización, mantenimiento, restauración de modos, etc., sino que se limita a utilizar las primitivas que ofrece la librería a tal efecto. Aunque a primera vista podría parecer una tarea superflua, sólo teniendo en cuenta que existen más de un centenar de tarjetas de vídeo diferentes en el mercado, - todas con arquitecturas y diseños totalmente diferentes, - podemos llegar a comprender la cantidad de trabajo inútil que la librería consigue evitar al desarrollador.

ARQUITECTURA

Por motivos de diseño, SVGALIB está dividida en dos partes claramente diferenciadas: VGALIB y VGAGL.

- VGALIB: su misión consiste en inicializar, mantener y dar acceso a los diferentes modos VGA, programando las diferentes tarjetas para tal efecto.
- VGAGL: proporciona al usuario primitivas básicas de dibujo, permitiéndole realizar tareas tales como pintar líneas, dibujar círculos, etc.

Para recalcar la diferencia entre ambas, los diseñadores de SVGALIB las han implementado como dos DLL's - librerías de enlace dinámico - independientes: libvga y libvgagl. Ambas poseen su propio juego de cabeceras, agrupadas respectivamente en los ficheros vga.h y vgagl.h.

INSTALACIÓN

Instalar SVGALIB en nuestro ordenador es una tarea francamente sencilla. A continuación vamos a describir el proceso que debemos seguir, suponiendo que vamos a colocar las fuentes de la librería a partir del directorio: /usr/src. Para poder operar en este directorio e instalar la librería en el sistema, será necesario entrar como superusuario, o bien adquirir sus privilegios mediante el comando su.

Esta librería gráfica proporciona un interfaz común de programación al usuario

En primer lugar tendremos que descomprimir el paquete - generalmente un fichero de tipo tgz - en el directorio de trabajo. Para ello utilizaremos el comando tar de la siguiente manera:

```
cd /usr/src
tar xvfz <path hasta la librería>/svgalib-1.2.10.tgz
```

Esta secuencia de comandos creará una jerarquía de directorios a partir del directorio de trabajo, en la cual se encuentra todo el código fuente de la librería. Ahora ya sólo nos queda compilarla e instalarla. Para ello bastará con introducir la orden:

```
make install
```

La cual se encargará de compilar el código fuente y colocar los diferentes componentes de la librería en sus directorios respectivos.

Una vez finalizado el proceso de instalación, deberemos advertir al sistema de la presencia de nuevas librerías dinámicas mediante la orden:

```
ldconfig
```

Si queremos cambiar el directorio de instalación de los diferentes componentes, deberemos modificar el fichero Makefile.cfg. Únicamente se deberán sustituir las referencias a los antiguos directorios por las nuevas ubicaciones de los ficheros que forman la distribución base.

VGALIB inicializa, mantiene y da acceso a los diferentes modos VGA y VGAGL da primitivas básicas de dibujo

Además de los archivos que hemos ido enumerando en la Tabla 1, podemos encontrar una serie de utilidades extra muy prácticas que pasamos a ver a continuación:

Demos

Las demos son ejemplos de utilización de la librería. Estos programas son muy útiles a la hora de ver cómo se utilizan diferentes primitivas de la librería. Uno de los más interesantes es el denominado speedtest. Permite ver la velocidad de la interfaz entre la tarjeta de vídeo y la memoria principal de nuestro ordenador. Este programa permite averiguar la rapidez de la tarjeta para cada una de las resoluciones posibles. Si queremos compilar las demos, bastará con teclear la orden:

```
make demoprogs
```

Speedtest permite ver la velocidad de la interfaz entre la tarjeta de vídeo y la memoria principal de nuestro ordenador

Los programas quedarán situados en los directorios:

```
/usr/src/svgalib-1.2.10/demo
/usr/src/svgalib-1.2.10/threeDkit
```

Utilidades de Texto

Además de las diferentes demos que vienen incluidas con la librería, existen una serie de utilidades que también forman parte del paquete. Permiten averiguar los contenidos de los registros de la tarjeta, restaurar un modo de texto que haya sido corrompido por un mal uso de la librería, etc. Para compilarlas basta con teclear:

```
make textutils
```

Quedando instaladas en el directorio en el que trabajamos:

```
/usr/src/svgalib-1.2.10/utils
```

Tabla 2. Listado de todos los modos gráficos que permite SVGALIB. Para poder utilizar un modo determinado es necesario que la tarjeta sea capaz de soportarlo.

Nombre	Anchura(X)	Altura(y)	Número de Colores
TEXT	na	na	na
G320x200x16	320	200	16
G640x200x16	640	200	16
G640x350x16	640	350	16
G640x480x16	640	480	16
G320x200x256	320	200	256
G320x240x256	320	240	256
G320x400x256	320	400	256
G360x480x256	360	480	256
G640x480x2	640	480	Monocromo
G640x480x256	640	480	256
G800x600x256	800	600	256
G1024x768x256	1024	768	256
G1280x1024x256	1280	1024	256
G320x200x32K	320	200	32768
G320x200x64K	320	200	65536
G320x200x16M	320	200	16.7 Millones
G640x480x32K	640	480	32768
G640x480x64K	640	480	65536
G640x480x16M	640	480	16.7 Millones
G800x600x32K	800	600	32768
G800x600x64K	800	600	65536
G800x600x16M	800	600	16.7 Millones
G1024x768x32K	1024	768	32768
G1024x768x64K	1024	768	65536
G1024x768x16M	1024	768	16.7 Millones
G1280x1024x32K	1280	1024	32768
G1280x1024x64K	1280	1024	65536
G1280x1024x16M	1280	1024	16.7 Millones
G800x600x16	800	600	16
G1024x768x16	1024	768	16
G1280x1024x16	1280	1024	16
G720x348x2	720	348	Monocromo
G320x200x16M32	320	200	16.7 Millones
G640x480x16M32	640	480	16.7 Millones
G800x600x16M32	800	600	16.7 Millones
G1024x768x16M32	1024	768	16.7 Millones
G1280x1024x16M32	1280	1024	16.7 Millones
G1152x864x16	1152	864	16
G1152x864x256	1152	864	256
G1152x864x32K	1152	864	32768
G1152x864x64K	1152	864	65536
G1152x864x16M	1152	864	16.7 Millones
G1152x864x16M32	1152	864	16.7 Millones
G1600x1200x16	1600	1200	16
G1600x1200x256	1600	1200	256
G1600x1200x32K	1600	1200	32768
G1600x1200x64K	1600	1200	65536
G1600x1200x16M	1600	1200	16.7 Millones
G1600x1200x16M32	1600	1200	16.7 Millones

UTILIZACIÓN

Para utilizar las librerías en nuestros programas, hemos de tener presentes dos cosas:

- Hemos de incluir las cabeceras, tanto de VGALIB como de VGAGL, añadiendo las líneas correspondientes a nuestro código. Estas líneas son:

```
#include <vga.h>
#include <vgagl.h>
```

- A la hora de compilar nuestros ficheros, deberemos tener cuidado a la hora de enlazar con las librerías dinámicas correspondientes. Para ello, bien en el fichero Makefile de nuestra aplicación o bien en la propia línea de comandos, añadiremos -lvga -lvgagl. Un ejemplo de compilación podría ser:

```
gcc ejemplo.c -o ejemplo -lvgagl -lvga
```

FUNCIONES BÁSICAS

Vamos a comenzar con el estudio de las funciones básicas de VGALIB, puesto que estas primitivas nos van a permitir colocar la tarjeta de vídeo en un estado determinado, interrogar al driver sobre las capacidades del modo elegido, averiguar qué tarjeta se encuentra instalada, etc. Estas funciones son:

- vga_init()
- vga_setmode()
- vga_hasmode()
- vga_getxdim()
- vga_getydim()
- vga_getcolors()
- vga_getmodeinfo()
- vga_getcurrentmode()
- vga_getdefaultmode()
- vga_getcurrentchipset()
- vvga_setchipset()
- vga_flip()

Función vga_init()

Esta función se encarga de inicializar la librería. Entre sus diferentes misiones destacan las siguientes:

- Detectar el chipset de la tarjeta de vídeo que está instalada en el PC.
- Reservar la memoria necesaria para la librería.
- Inicializar las diferentes variables relativas al driver del chipset detectado.
- Renuncia a los privilegios de superusuario para el resto del código.

Debe ser la primera función que se utilice en nuestro programa, ya que no se puede, - ni se debe, - utilizar ninguna otra primitiva de las librerías gráficas sin haberlas inicializado antes.

Esta función instala manejadores para la mayoría de las señales del sistema. Por defecto, si el usuario instala una señal antes de llamar a vga_init(), la librería devuelve al programa al modo texto y después llama a la rutina del usuario. Si por el contrario, instalamos el manejador después de llamar a vga_init(), la librería no captura la señal, ejecutándose únicamente nuestro procedimiento de atención.

VGALIB permitirá colocar la tarjeta de vídeo en un estado determinado e interrogar al driver por sus necesidades

Antes que nada es importante y necesario tener en cuenta y destacar que la librería gráfica VGALIB utiliza internamente las señales USR1 y USR2, por lo que nunca deberemos utilizarlas en nuestras aplicaciones.

Función vga_setmode()

Esta función se encarga de instalar el modo de vídeo que deseemos utilizar. La lista de los diferentes modos de vídeo se puede apreciar en la Tabla 2. Antes de llamar a esta función, puede ser muy útil comprobar si el modo está soportado mediante una llamada a la primitiva vga_hasmode().

Función vga_hasmode()

Esta función toma como argumento un modo determinado y pregunta al driver si la tarjeta va a ser capaz de conseguirlo. En caso de que la respuesta sea afirmativa, entonces devuelve el valor TRUE. En el caso contrario, el valor que obtendremos será FALSE.

Para instalar SVGALIB será necesario entrar como superusuario o bien adquirir sus privilegios

Es conveniente llamar a esta función antes de activar un modo, ya que las consecuencias de llevar a un dispositivo más allá de sus especificaciones pueden ser muy dolorosas, pudiendo llegar a tener que comprar una tarjeta de vídeo nueva.

Funciones vga_getxdim(), vga_getydim(), vga_getcolors()

Estas tres funciones son muy similares. Todas ellas solicitan al driver información acerca del modo de vídeo actualmente instalado:

- vga_getxdim(): devuelve la anchura de la pantalla en pixels.

- vga_getydim(): devuelve la altura del modo actual en pixels.
- vga_getcolors(): informa del número de colores diferentes que soporta el modo de gráfico actual.

Función vga_getmodeinfo()

Es una versión más completa de las tres funciones anteriores. vga_getmodeinfo() devuelve un puntero a una estructura donde se recogen todas las características del par Tarjeta/Modo actuales. La definición completa de la estructura se encuentra en el fichero de cabecera vga.h.

Función vga_getdefaultmode()

Esta primitiva permite obtener información acerca del modo de vídeo que el usuario tiene definido por defecto. Este modo se define mediante la variable de entorno GSVGAMODE, obteniéndose uno de los modos definidos en la Tabla 2. En el caso de que la variable de entorno no esté definida, devuelve el valor -1.

Función vga_getcurrentmode()

Mediante una llamada a esta función obtenemos el identificador del modo de vídeo que está actualmente instalado en la tarjeta de vídeo.

Función vga_getcurrentchipset()

Informa del tipo de tarjeta de vídeo que ha identificado la librería y cuyo driver esta siendo utilizado.

SVGALIB podría sustituir en el diseño gráfico a las X-Windows

Esta función puede utilizarse para excluir tarjetas que tengan problemas con nuestro software o que necesiten rutinas alternativas para realizar ciertas funciones.

Función vga_setchipset()

Permite especificar el driver que utilizará la librería. Para forzar la detección de una tarjeta determinada, deberemos incluir esta línea justo antes de la llamada a la primitiva vga_init(). Una lista completa de los diferentes drivers disponibles se puede encontrar en el fichero vga.h.

A partir de ahora podremos llegar a ver Linux como una plataforma posible para desarrollar juegos

Función vga_flip()

Esta función permite conmutar entre modo de vídeo y modo de texto, sin perder la información de ambas pantallas. Aunque su utilidad en el programa final es muy relativa, su empleo en el proceso de desarrollo es muy interesante. Mediante cambios de modo podemos simplificar la depuración de un programa de forma considerable.

CONCLUSIONES

Tras una breve introducción, en la que describimos los pros y los contras de Linux como una plataforma viable desde el punto de vista del desarrollo de juegos, analizamos las peculiaridades de la librería SVGALIB. En este estudio nos dimos cuenta de su atractivo especial, como sustituto de las X-Windows, a la hora del diseño de aplicaciones gráficas que requieran un acceso rápido a la memoria de vídeo, sin sobrecargas por parte de ningún tipo de protocolo.

Tras una descripción funcional de su arquitectura, describimos paso a paso el procedimiento de instalación del paquete completo de la librería, incluidos sus ejemplos.

Finalmente, describimos algunas de las primitivas más básicas que componen la librería. Fundamentalmente aprendimos a configurar la tarjeta para un modo gráfico determinado, siendo capaces de volver al modo de texto de una manera ordenada.

En posteriores artículos profundizaremos en el estudio de las diferentes posibilidades que nos ofrece SVGALIB, incluyendo primitivas de pintado, presentación de imágenes, manejo de periféricos auxiliares, etc.

1[1] Wintel acrónimo que define al Sistema Operativo Windows sobre la plataforma de Intel.

2[2] Raytracers: programas que generan imágenes mediante la proyección de rayos de luz sobre las distintas superficies de los objetos que componen una escena. Los efectos que se consiguen son extremadamente realistas, aunque requieren una gran cantidad de proceso.

3[3] IDE (Integrated Developement Environment): Entorno de Desarrollo Integrado, en el cual tenemos al compilador, el editor, el enlazador, el depurador y la información sobre el código en un entorno único. Un ejemplo del mundo Microsoft podría ser el Microsfot Visual C++.

4[4] RAD (Rapid Application Developement): Desarrollo Rápido de Aplicaciones. Herramientas que permiten realizar prototipos de aplicaciones de una manera rápida y fácil. Un ejemplo podría ser el Microsoft Visual Basic.

5[5] API: Aplication Program Interface. Conjunto de llamadas al sistema concebidas como un estándar, permite homogeneizar el acceso a un conjunto de servicios.

JAVA

Programación de Servlets (y III): su unión con JDBC

Juan Carlos Sastre Iglesias (jcsi@geocities.com)

La inmensa mayoría de las aplicaciones cliente-servidor del mundo real han de acceder a la información almacenada en bases de datos.
Los Servlets no escapan a esta realidad, puesto que se convierten en una parte fundamental de las arquitecturas cliente-servidor de tres capas en el entorno WWW.

La clave del éxito de las mejores páginas Web reside en la presentación de la información de forma interactiva, práctica y atractiva. La interactividad en la presentación de la información se basa en la consulta de bases de datos por parte de un software, que hace de intermediario entre el cliente y el sistema que almacena la información. A las aplicaciones cliente-servidor que utilizan este tipo de arquitectura se las denomina aplicaciones de tres capas.

ARQUITECTURA CLIENTE-SERVIDOR DE TRES CAPAS

En una aplicación cliente-servidor de tres capas, el software intermediario hace de puente entre la parte cliente y el servidor de base de datos que posee la información. De hecho, la aplicación cliente sólo se encarga de proporcionar una interfaz y de comunicarse con la parte intermedia, y es ésta última la que se encarga de realizar el trabajo de recuperación de información de la base de datos. Un esquema básico de esta arquitectura puede observarse en la figura 1. En el caso concreto que nos ocupa, la parte cliente sería el *browser* mostrando el formulario HTML, en el centro estaría el *Servlet*, que juntamente con JDBC (ver cuadro "JDBC"), interactuaría con la base de datos (figura 2). Finalmente, en la figura 3 puede observarse una posible implementación práctica de esta arquitectura. Por un lado tendríamos una maquina en la que un *browser* ejecuta un *applet* o muestra un formulario HTML. Por el otro tendríamos una máquina servidora, en la que un software servidor de Web recibe las peticiones de los clientes, y desencadena la ejecución del *Servlet*. Éste trabajaría con una base de datos alojada en otro servidor dedicado en exclusiva a esta función.

Los Servlets son básicos en la arquitectura cliente-servidor

Figura 1. Esquema básico de una aplicación estructurada en tres niveles.

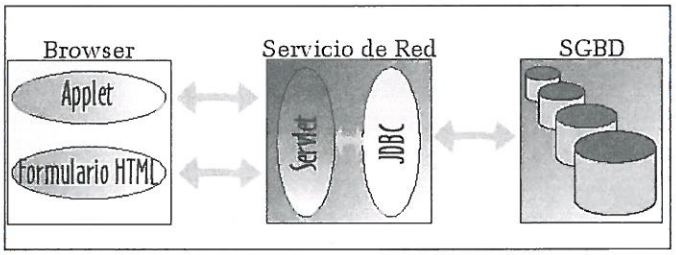


Figura 2. Aplicación de tres capas en la que la parte central está implementada utilizando Servlets y JDBC.

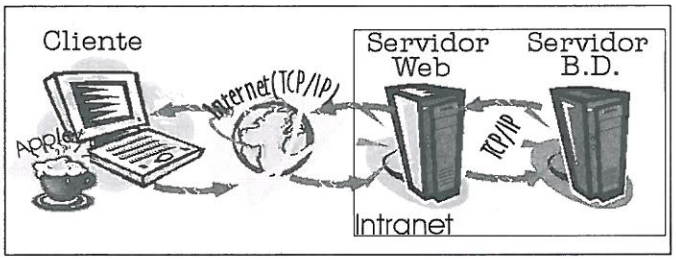


Figura 3. Ejemplo de una posible implementación real de una arquitectura de tres capas.

DESCRIPCIÓN DE LA APLICACIÓN EJEMPLO

Imaginemos que tenemos una página Web muy interesante, que recibe muchas visitas. Para conseguir que la gente que la visita participe de forma más activa, decidimos proponer una miniencuesta, con el reclamo de que entre todos los que participen se sorteará un magnifico premio. Para ello en nuestra página principal insertaremos un pequeño formulario, donde se deberá responder a una sencilla pregunta, e introducir los datos personales necesarios para que podamos contactar con el posible ganador del premio. Una vez el participante haya introducido los datos, y pulse el botón para mandarlos al servidor, recibirá un resumen con la estadística resultante hasta ese momento junto con un mensaje de agradecimiento.

En la página principal se deberá incluir un formulario

Una vez que hemos definido claramente nuestro objetivo, evaluaremos los elementos necesarios para alcanzarlo. En primer lugar se necesitará incluir en la página principal un formulario. Este consistirá en un grupo de opciones, de las cuales sólo se podrá seleccionar una, y una serie de campos de texto, donde el concursante introducirá sus datos personales. Finalmente, un par de botones, uno para enviar los datos al servidor, y otro para dejar en blanco todos los campos. En el listado 1 se encuentra el código fuente para este formulario. Finalmente en las figuras 4 y 5 puede observarse la entrada de datos por parte del usuario, y la posterior respuesta del servidor.

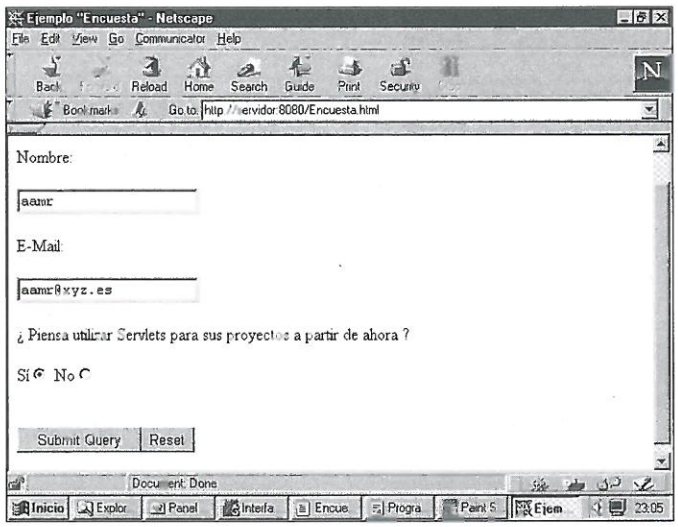


Figura 4. Formulario en el que el usuario ha de introducir información personal y la respuesta a la pregunta.

Listado 1. Código fuente.

```
<html>

 <head>
  <title>Ejemplo "Encuesta"</title>
 </head>

 <body>

 <form action=http://servidor:8080/servlet/Encuesta method=POST>

  <BR>
  <BR>Nombre:<BR>
  <BR>
  <input type=text name=NOMBRE>

  <BR>
  <BR>E-Mail:<BR>
  <BR>
  <input type=text name=EMAIL>

  <BR>
  <BR>ø Piensa utilizar Servlets para sus proyectos a partir de ahora ?<BR>
  <BR>
    SÌ<input type=radio name=RESPUESTA value=SI>
    No<input type=radio name=RESPUESTA value=NO>
  <BR>

  <BR>
  <BR><input type=submit><input type=reset></form>

 </body>

</html>
```

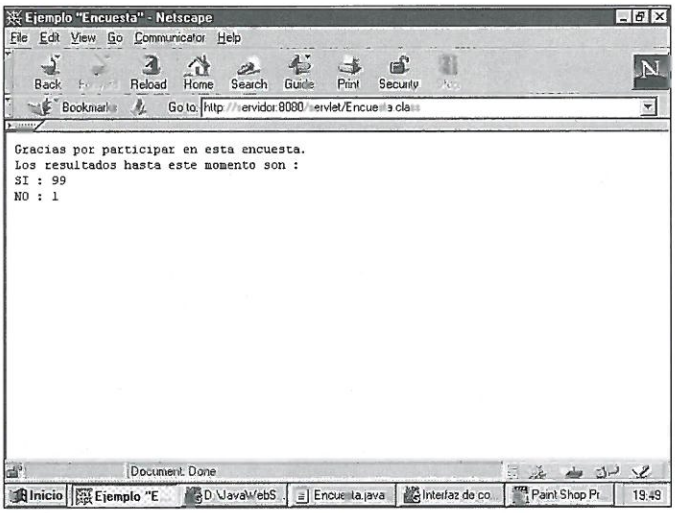


Figura 5. Tras procesar la información, el servlet responde al cliente con un mensaje, y el resultado de la estadística hasta ese momento.

En segundo lugar, se ha de crear el *Servlet*, cuya tarea consistirá en validar que el usuario introduzca todos los datos necesarios. Si todo está en regla, mostrará un mensaje de felicitación, junto con el resultado de la encuesta hasta ese momento. Si el usuario no ha introducido los datos correctamente, deberá avisarle, animándolo a que vuelva otra vez al formulario para introducir de la información requerida. En este capítulo no se implementará el código para proporcionar esta funcionalidad, porque nos centraremos en la interacción con la base de datos (se deja como ejercicio para el lector esta práctica). En la base de datos se almacenará la información del usuario, así como su respuesta a la pregunta.

ACCESO A LA BASE DE DATOS

Para poder ejecutar el ejemplo necesitamos crear una pequeña base de datos, y configurar el sistema para que podamos acceder a ella desde nuestros programas. Para la comunicación se utilizará ODBC. Los lectores que no dispongan del ODBC instalado en su sistema Windows, pueden utilizar Microsoft Data Access Pack, que se encuentra en el CD-ROM de la revista.

Una vez que se disponga de ODBC instalado en el sistema, será necesario llevar a cabo una fase de configurarción. Si pulsamos sobre el icono ODBC que se encuentra en el Panel de Control, aparecerá un cuadro de diálogo como similar al mostrado en la figura 6. Seleccionaremos Text Files, ya que así no se requerirá ningún software adicional de base de datos. Al pulsar sobre el botón Addnos aparecerá una lista donde deberemos seleccionar el driver que deseamos utilizar (figura 7). Seleccionaremos Microsoft Text Driver, y pulsaremos el botón Finalizar. Tan sólo queda asignar un nombre identificativo, que será utilizado por nuestros programas para trabajar con la base de datos, y seleccionar el directorio donde se encuentran los ficheros (figura 8). De esta forma queda definida la configuración para nuestra pequeña base de datos, que se basará en simples ficheros de texto.

Con la función Addnos obtendremos el driver

Al no utilizar ningún software de bases de datos, no podremos utilizar un

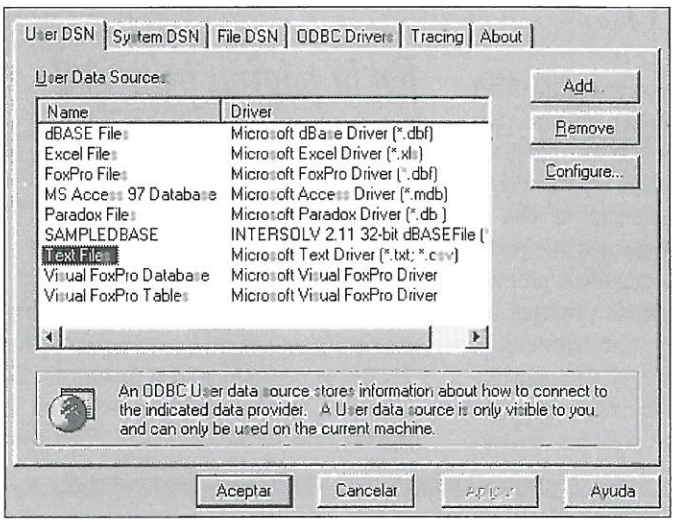


Figura 6. En el diálogo de ODBC seleccionaremos una base de datos basada en ficheros de texto.

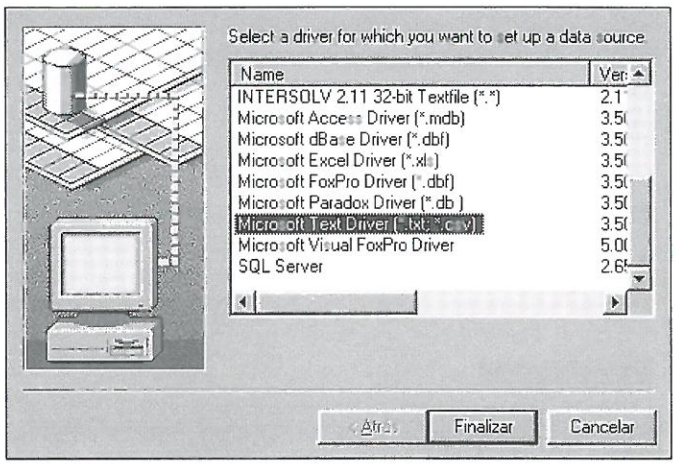


Figura 7. Es esta parte de la configuración del ODBC seleccionaremos el driver que utilizar.

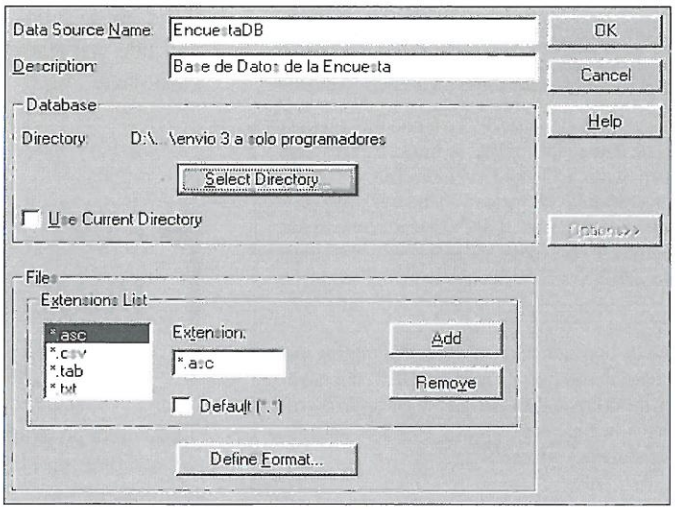


Figura 8. Debemos asignar un identificador a la base de datos que será nuestra fuente de información.

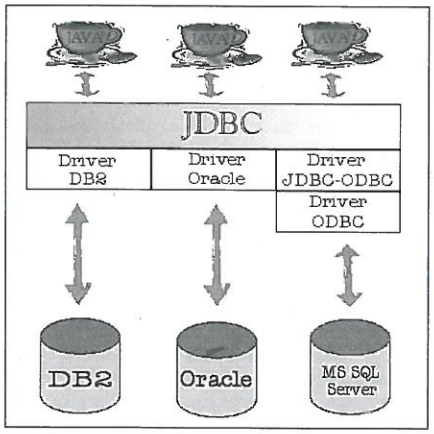


Figura 9. Los drivers JDBC pueden comunicarse directamente con las bases de datos.

asistente para crear las tablas necesarias. Es por esto que realizaremos un pequeño programa para crear la tabla que necesitamos (listado 2). Utilizando el driver JDBC-ODBC se someterá la sentencia SQL a la fuente de datos ODBC. Si ejecutamos el programa, podremos observar que se ha creado dos archivos. Uno es schema.ini, que contiene información de definición, y el otro es BaseDatosEncuesta, que es la tabla que utilizaremos para almacenar la información.

Obviamente, en aplicaciones profesionales, que no es el caso que nos ocupa, se utilizan bases de datos como Oracle ó DB/2 de IBM, que proporcionan una serie de prestaciones indispensables para en-

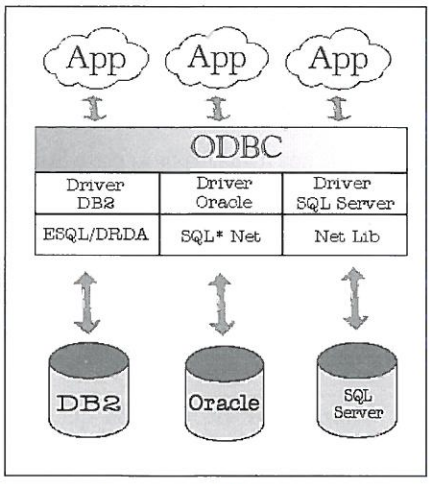


Figura 10. Los drivers ODBC no se comunican directamente con las bases de datos, lo que ofrece un rendimiento inferior con respecto a JDBC.

Vendor	Type	DBMSs supported
Agave Software Design	3	Oracle, Sybase, Informix, others via ODBC
Asgard Software	3	Unisys A series DMSII database
Borland	4	InterBase 4.0
Caribou Lake Software	3	Ingres
Connect Software	4	Sybase, MS SQL Server
DataRamp	3	Several dozen through ODBC drivers.*
Ensoden, Inc.	3	Several through ODBC.*
IBM	2/3	IBM DB2 Version 2
IBM	4	DB2 for OS/400
JWE Technologies	4	mysql
IDS Software	3	Oracle, Sybase, MS SQL Server, MS Access, Informix, Watcom, and others via ODBC
I-Kinetics, Inc.	3	Oracle, Informix, Sybase, and others via ODBC
Imaginary	4	mSQL
InterSoft	3	Essentia
Intersolv	3	DB2, Ingres, Informix, Oracle, Microsoft SQL Server, Sybase 10/11
JavaSoft	1	Several dozen through ODBC drivers.*
KonaSoft, Inc.	3/4	Sybase, Oracle, Informix
NetAway	3	Oracle, Informix, Sybase, MS SQL Server, DB2, others via ODBC
OpenLink	3	Oracle, Informix, Sybase, MS SQL Server, CA-Ingres, Progress, Unify, PostgreSQL, Solid, and others via ODBC.
Oracle Corporation	2/4	Oracle
SAS Institute Inc.	3/4	SAS, and via SAS/ACCESS, Oracle, Informix, Ingres, and ADABAS
SCO	3	Informix, Oracle, Ingres, Sybase, Interbase
StormCloud Development	3	Any DBMS that is accessible via ODBC
Sybase, Inc.	3/4	Sybase SQL Server, SQL Anywhere, Sybase IQ, Replication Server and more than 25 enterprise and legacy datase servers via Sybase OmniCONNECT
Symantec	3	Oracle, Sybase, MS SQL Server, MS Access, Watcom and others via ODBC
Trifox, Inc.	3	ADABAS, DB2, Informix, Ingres, Oracle, Rdb, SQL Server, Sybase, and legacy systems via GENESIS.
Visigenic	3	Several dozen through ODBC drivers.*
WebLogic	2	Oracle, Sybase, MS SQL Server
WebLogic	3	Several dozen through ODBC drivers.*
XDB Systems, Inc.	1/3	Many databases through ODBC.*
Yard Software GmbH	4	YARD-SQL Database

Figura 11. Lista de drivers JDBC disponibles en el mercado, junto con su tipo basado en la clasificación de Javasoft.

Cuadro 1. JDBC.

JDBC son las siglas de Java DataBase Connection, y se trata de una API Java que proporciona una interfaz similar a ODBC para trabajar con bases de datos SQL. JDBC se basa en el X/Open SQL CLI, al igual que el ODBC de Microsoft. X/Open SQL CLI (Call Level Interface) es un estándar definido por el SAG (SQL Acces Group es un consorcio de varios vendedores), que proporciona un visión unificada para el acceso remoto a bases de datos. JDBC fue desarrollado por JavaSoft, Sybase, Informix, IBM y otros, y se puede utilizar para trabajar con todos los Sistemas Gestores de Bases de Datos (SGDB) SQL que existen: DB2, Oracle7, Sybase, SQL Server, SQLBase, XDB, etc.

Conviene recalcar que JDBC no es un lenguaje de consulta, sino que simplemente es una interfaz basada en Java para trabajar con SQL. Las aplicaciones pueden utilizar JDBC, por ejemplo, para someter sentencias SQL a un sistema gestor de bases de datos. JDBC se encarga de gestionar aspectos tales como conexiones a bases de datos, recuperación de resultados de las consultas, transacciones, conversiones de tipos SQL a tipos Java y viceversa.

JDBC vs ODBC

JDBC tiene frente a ODBC varios puntos a favor que conviene resaltar:

1) ODBC es una interfaz definida en C, mientras que JDBC es una interfaz definida en Java. Por ello ODBC es una API basada en procedimientos que hace un uso intensivo de punteros, mientras que JDBC es una interfaz orientada al objeto. Es decir, mientras en ODBC se trabaja con manejadores (handlers) y llamadas a las API, en JDBC se trabaja con objetos (o clases), interfaces y métodos.

2) ODBC define una serie de funciones que se dividen en tres niveles de conformidad, cada uno de los cuales incluye un grupo de éstas. El problema es que no todos los drivers existentes cumplen por igual estos niveles, y por tanto las aplicaciones han de tener en cuenta el nivel de conformidad del driver con el que están trabajando para saber que funciones pueden utilizar. Por el contrario, todo driver JDBC ha de pasar una serie de tests para confirmar que cumple con el ANSI SQL92 Entry Level, que define el mínimo común denominador entre cualquier SGDB SQL del mercado. Esto quiere decir que debe implementar todas las clases e interfaces abstractas que define el paquete java.sql, lo cual nos garantiza que cualquier aplicación que escribamos utilizando JDBC podrá trabajar con cualquier SGBD del mercado sin limitaciones.

3) Los drivers ODBC han de trabajar junto con los drivers nativos del sistema gestor de base de datos, mientras que JDBC no (figuras 9 y 10).

Drivers JDBC

JavaSoft clasifica los drivers JDBC en cuatro categorías :

Tipo 1 :
Estos drivers utilizan tecnología de puente para conectar con las bases de datos. El driver JDBC-ODBC que utilizamos en este capítulo es un buen ejemplo. Simplemente proporciona una pasarela hacía la API ODBC.

Tipo 2:
El código Java de estos drivers, contiene llamadas a métodos nativos en C ó C++, proporcionados por los vendedores de las bases de datos.

Tipo 3:
Este tipo de drivers utiliza sockets para conectarse con un software middleware, que es el que trabaja con la base de datos.

Tipo 4:
Se comunican directamente con la base de datos utilizando sockets. Son la solución más pura desde el punto de vista Java. Este tipo de drivers sólo los puede proporcionar el mismo vendedor de la base de datos.

El mercado de drivers JDBC

En la figura11 se muestra una lista con los drivers que existen en el mercado. Asimismo se incluye información sobre el tipo en que se pueden encuadrar.

torno empresariales, y que cada vez tienen una orientación mayor hacia el lenguaje Java.

¿ CÓMO SE PROGRAMA CON JDBC ?

Tanto en el programa para crear la base de datos, como en el Servlet que crearemos posteriormente, se hace uso intensivo del paquete java.sql, que es el que se corresponde con JDBC. Tomando como base la aplicación de este capítulo, los pasos que seguir para interactuar con la base datos serían los siguientes :

1) Cargar el driver que se va utilizar. Por ejemplo, para indicar que vamos a utilizar el driver puente JDBC-ODBC, escribiríamos una línea como la siguiente:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

2) Establecer una conexión con la base de datos. En nuestro caso:

```
String url = "jdbc:odbc:EncuestaDB";
String usuario = "";
String contrasena = "";
Connection conexion =
      DriverManager.getConnection ( url, usuario,
      contraseña);
```

3) Crear una instancia de la clase Statement a partir de la conexión. De esta forma podremos someter las sentencias SQL a la base de datos:

```
Statement mandato =
      conexion.createStatement();
```

4) Utilizaremos los métodos que nos proporciona la clase Statement para someter la sentencia SQL:

```
String strSentenciaSQL = " ..... ";
ResultSet resultado =
      mandato.executeQuery(strSentenciaSQL);
```

5) Para trabajar con los resultados utilizamos la clase ResulSet, que proporciona los métodos necesarios para efectuar la tarea. Un ejemplo sería:

```
String strSentenciaSQL = "SELECT Nombre
      FROM BaseDatosEncuesta;";
ResultSet resultado =
      mandato.executeQuery(strSentenciaSQL);
while(resultado.next())
{
  String strNombre =
      resultado.getString("Nombre");
}
```

6) Finalmente sólo queda cerrar la conexión con la base de datos:

```
conexion.close();
```

CÓDIGO DEL SERVLET

La mécanica de trabajo para utilizar JDBC con los servlets seguirá siempre un estilo similar al del listado 3, que se corresponde al código fuente de nuestra aplicación.

En el método init se efectúa todo el trabajo que tiene que ver con la inicialización (pasos 1, 2 y 3 del apartado anterior). En primer lugar se registra el driver, seguidamente se establece la conexión y finalmente se obtiene una instancia de Statement. Las instancias de Connection y Statement deberán ser visibles por todo el código del Servlet.

En el método service se trabaja propiamente con la base de datos. Utilizando instancias de Statement y ResultSet se someten sentencias SQL para añadir, modificar, eliminar y consultar registros de la base de datos.

Finalmente cuando llega la hora de descargar el Servlet de memoria, se ejecuta el método destroy. En este método se finalizará la conexión con la base de datos, mediante el método close de la clase Connection.

Listado 2. Tabla de base de datos.

```
//
// CrearEncuestaDB.java
//

import java.sql.*;

public class CrearEncuestaDB
{

 public static void main(String[] args)
 {

  String URL      = "jdbc:odbc:EncuestaDB";
  String usuario  = "";
  String contrasena = "";

  try
  {
   Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  }
  catch (Exception e)
  {
   System.out.println("Error al cargar el driver JDBC/ODBC.");
   return;
  }

  Statement  mandato  = null;
  Connection conexion = null;

  try
  {
   conexion = DriverManager.getConnection ( URL, usuario, contrasena);
   mandato  = conexion.createStatement();
  }
  catch (Exception e)
  {
   System.err.println("problemas al conectar con "+URL);
  }

  try
  {
   mandato.execute("CREATE TABLE BaseDatosEncuesta (Nombre VARCHAR (60), Email
VARCHAR (60), Respuesta VARCHAR (2));");
   conexion.close();
  }
  catch (Exception e)
  {
   System.err.println("Problemas al mandar mandato SQL a "+URL+": "+e.getMessage());
  }

 }

}
```

Listado 3. Código fuente de la aplicación.

```
//
//  Encuesta.java
//

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class Encuesta extends HttpServlet
{

 Statement  mandato  = null;
 Connection conexion = null;

 public void init(ServletConfig config) throws ServletException
 {

  super.init(config);

  String URL      = "jdbc:odbc:EncuestaDB";
  String usuario  = "";
  String contrasena = "";

  try
  {
   Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  }
  catch (Exception e)
  {
   System.out.println("Error al cargar el driver JDBC/ODBC.");
   return;
  }

  try
  {
   conexion = DriverManager.getConnection ( URL, usuario, contrase-
na);
   mandato  = conexion.createStatement();
  }
  catch (Exception e)
  {
   System.out.println("problemas al conectar con "+URL);
 return;
  }

 }


 public void service( HttpServletRequest  peticion,
                      HttpServletResponse respuesta
)
  throws ServletException, IOException
 {

  /* creaciÛn del flujo de salida hacia el cliente */

  ServletOutputStream out = respuesta.getOutputStream();
  respuesta.setContentType("text/html");

  /* recuperamos los valores que nos manda el cliente */

  String strNombre   = peticion.getParameter("NOMBRE");
  String strEmail    = peticion.getParameter("EMAIL");
  String strRespuesta = peticion.getParameter("RESPUESTA");

  /* insertamos los datos en la base de datos */

  try
  {
     mandato.executeUpdate("INSERT INTO BaseDatosEncuesta VA-
LUES( '"+strNombre+"', '"+strEmail+"', '"+strRespuesta+"');");
  }
  catch(java.sql.SQLException e)
  {
   System.out.println(e);
 return;
  }

  /* leemos todos los registros para crear la estadÌstica */

  try
  {
   int intSI = 0;
   int intNO = 0;
    ResultSet resultado = mandato.executeQuery("SELECT Respuesta
FROM BaseDatosEncuesta;");
   while(resultado.next())
 {
    String resp = resultado.getString("Respuesta");
    if(resp.compareTo("SI")==0) intSI++;
    else intNO++;
   }
   out.println("Gracias por participar en esta encuesta.");
   out.println("<BR>Los resultados hasta este momento son :");
   out.println("<BR>          SI : "+intSI);
   out.println("<BR>          NO : "+intNO);
  }
  catch(java.lang.Exception e)
  {
   System.out.println(e);
 return;
  }

 }

 public void destroy()
 {
  try
  {
   conexion.close();
  }
  catch(java.sql.SQLException e)
  {
   System.out.println(e);
 return;
  }
 }

}
```

PROGRAMACIÓN DE SOCKETS

Integración de CGI en el servidor web (III)

Enrique de la Lastra (elastra@redestb.es)

INTRODUCCIÓN

En los dos artículos anteriores hemos creado un servidor web que cumplía con la función básica de un servidor de esta clase: servir páginas web (HTML). Sin embargo, el servidor está algo incompleto, ya que no permite a los navegadores (clientes web) que le envíen datos de consultas personalizadas o datos de actualización.

Para conseguir que el cliente envíe datos al servidor, debemos dar soporte al comando *POST* y generar el código necesario para ejecutar la aplicación de servidor cuyo nombre acompaña a la petición. Las aplicaciones que nuestro servidor será capaz de ejecutar y con las que podrá establecer un diálogo, serán programas *CGI* (también conocidos como "secuencias *CGI*" por su implementación en lenguajes de *scripts* en las máquinas UNIX). En concreto, seguirán la especificación *WinCGI* que establece los mecanismos para proveer *CGI* sobre máquinas Windows.

En este artículo, vamos a investigar en qué consiste la interfaz *CGI* y en particular haremos un análisis de la última especificación de *WinCGI*, que será la que emplearemos como interfaz *CGI* en nuestro servidor web. Para que la explicación no quede en mera información, haremos un programa *WinCGI* con Delphi plenamente funcional en cualquier servidor web que cumpla con dicha interfaz.

CGI

CGI ó *Common Gateway Interface* (Interfaz de Puerta Común) consiste en una especificación que permite conectar una aplicación a un servidor web. Define todas las reglas necesarias que deben cumplir tanto el servidor como las aplicaciones que dialoguen con él. De esta manera, al cumplir estas reglas, el servidor podrá comunicarse con cualquier programa que también se adapte a ellas. Así cualquier servidor que soporte la interfaz *CGI*, podrá comunicarse con cualquier programa *CGI* y cualquier programa *CGI* podrá, a su vez, funcionar en conjunto con cualquier servidor.

La CGI ó Common Gateway Interface (Interfaz de Puerta Común), es una especificación que permite conectar una aplicación a un servidor web

Como la *CGI* sólo define una interfaz, no entra en la especificación del lenguaje de programación con el que se deben crear los programas *CGI*. Sólo indica

Si nuestra meta es ampliar el diálogo con un servidor web, de forma que consigamos que el cliente pueda además de recibir información, enviar datos al servidor, tendremos que incorporarle una interfaz con aplicaciones residentes que amplíen su funcionalidad.
A lo largo de este artículo iremos viendo la especificación CGI.

la forma en que debe realizarse la comunicación con la interfaz, pero no define cómo debe implementarse, ni sobre qué sistema operativo. Esto abre inmensas posibilidades ya que, en la práctica, se puede utilizar desde cualquier lenguaje interpretado del estilo de *Perl* en máquinas *UNIX* a cualquier lenguaje de programación compilado del estilo de Delphi en máquinas *Intel* bajo Windows.

El funcionamiento de la interfaz CGI en la práctica, sigue las siguientes pautas:

1. El cliente lanza una petición a una URL, que en lugar de referirse a una página HTML, hace referencia a un programa ejecutable. Por ejemplo, introduce la URL: http://www.towercom.es/cgi/busca_datos?nombre=elastra donde busca_datos es un ejecutable en el servidor y nombre=elastra, un dato que se pasará como parámetro.

2. El servidor web (que atiende las peticiones dirigidas a www.towercom.es) comprueba que busca_datos no es una página sino un ejecutable.

3. De la propia petición, el servidor web extrae todos los datos que el cliente envía y que serán necesarios para ejecutar la aplicación.

4. El servidor web ejecuta el programa busca_datos transmitiéndole todos los parámetros.

5. El programa busca_datos se encarga de consultar en una base de datos, y de acuerdo a los parámetros de la petición, devuelve la información solicitada.

6. El programa busca_datos genera una respuesta en la que se incluyen tanto los códigos de respuesta HTTP como la página HTML con los datos solicitados.

7. Esta respuesta se le pasa al servidor web, que será el que efectivamente se la envíe al cliente.

Se puede apreciar cómo la interacción real se produce entre el cliente y el servidor web y entre el este último y proceso CGI, aunque la comunicación "lógica" se produce entre el cliente web y el programa CGI residente en el servidor.

WINCGI

La especificación CGI fue realizada pensando en máquinas UNIX. En éstas, la información CGI del servidor web se suministraba a través de la entrada/salida estándar y de variables de entorno. Sin embargo hay una serie de características que la hacen inutilizable en máquinas Windows (por ejemplo el manejo de stdin/stdout). Es por ello que los creadores del servidor Website (para Windows), transformaron la especificación para adaptarla a los ordenadores con entorno Windows. A esta nueva especificación se la denominó WinCGI. Con ella, en lugar de la entrada y salida estándar (stdin/stdout) se utilizan ficheros de intercambio, lo que permite programar aplicaciones con cualquier compilador: Delphi, Visual Basic, C++, etc. Actualmente su versión oficial es la 1.3.a.

La especificación WinCGI se hizo pensando en las máquinas Windows

La línea de comandos que el servidor web empleará para ejecutar una aplicación CGI (WinCGI, en realidad), constará del nombre del programa CGI ejecutable y de un nombre de fichero con extensión ini, que servirá de archivo de intercambio entre el servidor web y el programa CGI. Esto quiere decir que será una línea del estilo:

```
C:\cgi-win\aplicacion_cgi.exe
    C:\temp\fichero_de_intercambio.ini
```

Un servidor web y un programa WinCGI se comunican a través de un fichero ".ini"

La aplicación aplicacion_cgi.exe será el programa CGI que invoca el servidor web ante una petición de un cliente, y el archivo fichero_de_intercambio.ini será el archivo que el servidor y el programa CGI utilizarán para informar y ser informados, respectivamente, de la petición que haga el cliente.

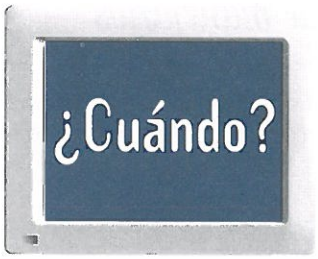
El hecho de que al realizar la especificación de WinCGI los autores se inclinasen por utilizar un archivo con extensión ini como archivo de intercambio, se debe la facilidad de manejo de este tipo de archivos en entornos Windows, ya que la propia API de Windows dispone de funciones de acceso directo a cualquier línea de cualquier sección de un fichero de inicialización. En nuestro caso esta labor será incluso más fácil, ya que existe una clase en Delphi (TIniFile), especializada en la lectura y escritura de este tipo de ficheros, y que convierte el manejo de los mismos en algo trivial.

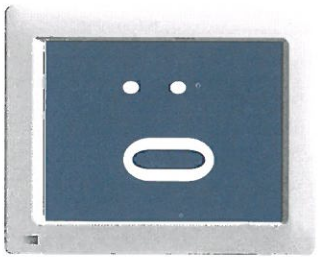
El fichero ini es pues la parte estándar a la que deberán acomodarse tanto el servidor como el programa CGI si quieren entenderse. Es decir, no podremos inventarnos las secciones de este archivo, sino que éstas deberán seguir unas normas concretas.

El fichero ".ini" consta de una serie de secciones estándares para Servidor y Programa

En concreto este fichero consta de ocho secciones, con los siguientes títulos: Accept, System, ExtraHeaders, FormLi-

INTERNET ASEQUIBLE





teral, FormExternal, FormHuge y FormFile. Cualquier dato de intercambio debe ir insertado en una de esas ocho secciones.

Vemos estas secciones una a una, con sus posibles entradas y valores. Cada elemento de cada sección, se muestra en la forma entrada=valor. El valor será siempre una cadena (string), y si esta cadena está vacía, la entrada se omite.

SECCIÓN [CGI]

Esta sección contiene la identificación del programa CGI e información de acceso.

1. CGI Version. Version de la especificación WinCGI utilizada por el servidor.
2. Query String. Si la URL contiene el signo ? entonces, en esta entrada se guarda la cadena que sigue a dicho signo (que contendrá los parámetros de la consulta). El servidor debe suministrar este valor tal cual lo envió el cliente, quedando bajo la responsabilidad del programa CGI, la descodificación del mismo en las partes clave=valor.
3. Server Name. Nombre de dominio del servidor (para poder reconstruir la URL completa).
4. Server Port. Puerto a través del cual está escuchando el servidor.
5. Server Software. Nombre y versión del servidor web.
6. Referer. La URL de la página web que contiene el enlace al programa CGI (esta entrada puede no estar presente si el enlace fue escrito con el teclado o activado a través de un bookmark).
7. Request Method. Método de petición (GET ó POST) utilizado. El método POST suele utilizarse cuando los datos que deben transmitirse son tan largos que el método GET no puede manejarlos.
8. Request Protocol. Protocolo y versión, a través del cual se ha recibido la petición.
9. Remote Address. Dirección IP del cliente.
10. Remote Host. Nombre de dominio del cliente.
11. Content File. Ruta completa del fichero con los datos que envió el cliente (y que ha rellenado el servidor con esos datos).
12. Content Length. Longitud de los datos enviados.
13. Content Type. Tipo MIME del contenido de los datos
14. Executable Path. Ruta del fichero WinCGI que está siendo ejecutado.

POST se utiliza cuando los datos que deben transmitirse son tan largos, que GET no puede manejarlos

A continuación se muestra un ejemplo de cómo podría ser esta sección :

```
[CGI]
CGI Version=WinCGI/1.3b
Referer=http://www.towercom.es/cgi.htm
Query String=articulo=Web Server 1/3
Server Name=www.towercom.es
Server Software=WebSitePro/2.0.36e
Server Port=80
Remote Host=http://www.towercom.es/
Remote Address=192.168.234.123
Request Protocol=HTTP/1.0
Request Method=POST
Content File=fichero_de_datos.dat
Content Length=2534
Content Type= multipart/form-data
Executable Path=c:\home\cgi-win\cgi_01.exe
```

SECCIÓN [ACCEPT]

Esta sección contiene la lista de los tipos de datos que acepta el cliente. La sintaxis será: tipo/subtipo=valor. Si el valor no se incluye, se entiende por defecto que es yes (si). Por ejemplo, esta sección podría ser como la siguiente:

```
[Accept]
image/gif=yes
image/x-xbitmap=yes
image/jpeg=yes
image/pjpeg=yes
*/*=yes
```

SECCIÓN [SYSTEM]

La sección System contiene información específica de la implementación del CGI.

1. GMT Offset. Tiempo (en segundos) que es necesario añadir a la hora GMT (Greenwich Mean Time, Hora Respecto al Meridiano Greenwich) para convertirla a la hora local.
2. Debug Mode. Informa si el servidor está en modo "depuración".

Output File nos da la ruta completa del fichero en el que el servidor web recibirá los resultados del programa CGI

SÍ, ASEQUIBLE

NO ENCONTRARÁS UNA OFERTA MEJOR

¿Cómo?

Con un completo curso multimedia que te explicará desde en qué consiste el correo electrónico hasta qué es el FTP. Y, con conexión gratuita a Internet hasta el 31 de diciembre de 1.998.

¿Cuánto?

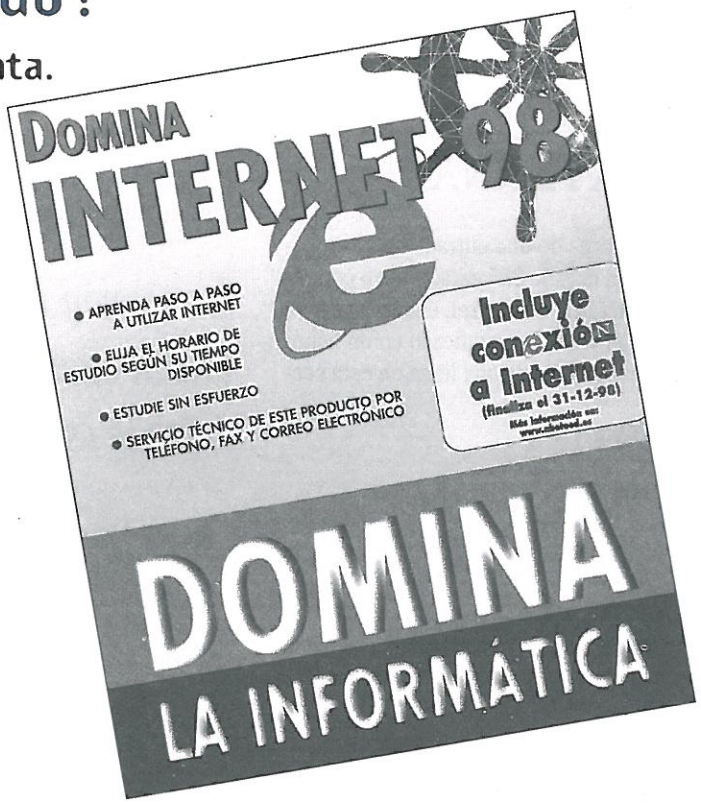
Por sólo 3.995 ptas. Iva incluido.

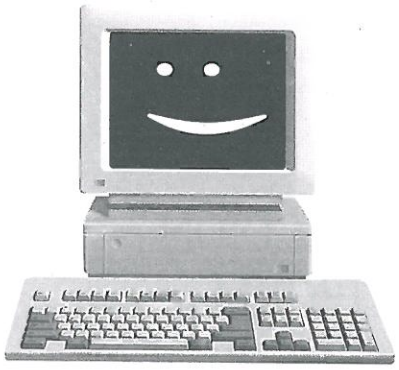
¿Dónde?

Podrás encontrarlo en librerías, quioscos, tiendas de informática, Vips, Crisol, Fnac, El Corte Inglés, Casa del libro o en el teléfono 91-6614211.

¿Cuándo?

Ya a la venta.





ABETO editorial

C/ Aragoneses, 7
28108 Pol. Ind. Alcobendas (MADRID)
Tel.: (91) 661 42 11* Fax: (91) 661 43 86
http://www.abetoed.es

También versión sin conexión por 2.995 ptas.

3. Output File. Ruta completa del fichero en el que el servidor web espera recibir los resultados del programa CGI.

4. Content File. Ruta completa del fichero que almacena el contenido de la petición.

Por ejemplo, esta sección podría ser como la siguiente:

```
[System]
GMT Offset=-3600
Debug Mode=no
Output File=c:\home\cgi-temp\salida_cgi.out
Content File=c:\home\cgi-temp\salida_cgi.con
```

SECCIÓN [EXTRA HEADERS]

Cabeceras "extra" incluidas en la petición (ver en el artículo anterior la especificación del HTTP). Por ejemplo:

```
[Extra Headers]
User Agent=Mozilla/4.03 [es] (Win95; I)
Connection=Keep-Alive
Pragma=no-cache
Host=localhost
```

SECCIÓN [FORM LITERAL]

Si se usa el método POST en la petición y ésta proviene a su vez de un formulario, en esta sección se escribirán las entradas correspondientes a las entradas del formulario HTML.

Los datos vendrán en la URL en la forma "entrada1=valor1&entrada2=valor2..." El servidor deberá dividir esta sentencia en pares "entradaX=valorX" y reemplazar los signos + por espacios en blanco (ya que los espacios en blanco se transmiten de esta forma en la URL).

La ruta del fichero que almacena el contenido de la petición está en Content File

Esas entradas son las que el servidor web introducirá en el fichero de extensión ini. Por ejemplo, esta sección podría ser como la siguiente:

```
[Form Literal]
Articulo1=ServidorWeb2.htm
Articulo2=ClienteEcho.htm
Nombre=Enrique de la
Apellidos=Lastra
Email=elastra@redestb.es
```

SECCIÓN [FORM EXTERNAL]

Si los valores de una entrada del formulario tienen más de 255 caracteres o contienen caracteres de control, el servidor web escribe el valor descodificado en un fichero temporal y añade una línea en esta sección.

Tanto el servidor como el programa CGI deberán acomodarse al fichero ini

Esta sección se podría parecer a lo siguiente:

```
[Form External]
field300chars=C:\home\cgi-temp\HS19AF6C.000
    300
fieldwithlinebreaks=C:\home\cgi-
    temp\HS19AF6C.001 43
```

SECCIÓN [FORM HUGE]

Esta sección amplía a la anterior, y se utiliza cuando el valor de una entrada es mayor de 64K. En ese caso, el servidor no descodifica el valor, pero añade una línea en esta sección, en la que se indica el fichero de contenido donde aparecerá dicho valor y la posición del primer byte del valor, en dicho archivo. Por ejemplo:

```
[Form Huge]
field230K=c:\home\cgi-temp\HS19AF6C.002
    276920If
```

SECCIÓN [FORM FILE]

Esta sección se utiliza cuando la petición contiene varios uploads y no la utilizaremos en ningún caso.

La sección Form Huge se utiliza cuando el valor de entrada es mayor de 64K

En el próximo número crearemos un programa basado en la interfaz WinCGI, de tal forma que con un formulario HTML (donde aparecerá la referencia al programa WinCGI, se podrá acceder a un determinado artículo de la serie "Creación de un servidor Web". Se indicarán los pasos para construir el proyecto en Delphi, además de incluir las páginas HTML, el código final del programa y el ejecutable final.

PROGRAMACIÓN CIENTÍFICA

Representación gráfica de funciones tridimensionales

Manel Guerrero (tarom@iname.com)

En este artículo aprenderemos todo lo necesario para crear aplicaciones capaces de procesar y representar gráficamente funciones tridimensionales. Podremos crear un programa mediante el cual seamos capaces de introducir funciones para luego lograr visualizarlas utilizando el teclado.

EVALUACIÓN DE FUNCIONES MATEMÁTICAS

En realidad, el objetivo no consiste en crear un programa que sólo sea capaz de representar gráficamente una función determinada, o una serie limitada de funciones. Se trata de crear un programa mediante el cual el usuario pueda introducir la función que desee visualizar utilizando el teclado.

Por lo tanto antes de poder representar gráficamente una función, sea del tipo que sea, nuestro programa deberá ser capaz de procesarla de forma que pueda trabajar con ella. Es decir, deberá convertir la secuencia de caracteres introducidos por el usuario en una estructura de datos que sea capaz de manejar con eficiencia.

El programa procesará la imagen para trabajar

En los siguientes apartados estudiaremos la serie de transformaciones que debemos realizar para transformar la cadena de caracteres, que representa la expresión que se desea representar, en una estructura de datos que permita calcular valores de la función en cuestión.

LA PRIORIDAD ENTRE OPERADORES

Cuando se escribe una expresión matemática usamos paréntesis para indicar el orden en que deberán realizarse las operaciones. Para no trabajar con expresiones con demasiados paréntesis, lo que podría dificultar su legibilidad, se estableció el orden de prioridad entre los diferentes operadores matemáticos que todos conocemos. (Figura 1a) De esta forma, solamente es necesario el uso de los paréntesis cuando el orden en que deben realizarse las operaciones no es el indicado por las reglas de prioridad.

Por lo tanto, la primera de las transformaciones que deberemos realizar a la cadena de caracteres de entrada consiste en colocar todos los paréntesis que normalmente obviamos. (Figura 1b)

LA NOTACIÓN POSTFIJA

Una vez que se dispone de la expresión con todos los paréntesis correspondientes, el siguiente paso consistirá en pasar la expresión (que estaba en notación infija) a notación postfija. Antes de nada

Orden	Operadores
1º	Potencia y raíz cuadrada
2º	Multiplicación y división
3º	Suma y resta

Figura 1a. Orden de prioridad entre operadores binarios.

(a+b)*c^d ⇒ ((a+b)*(c^d))

Figura 1b. Añadir los paréntesis obviables.

recordaremos cuáles son los tres tipos de notación para expresiones matemáticas (atendiendo a la posición en que se sitúan los operadores respecto de sus operandos): (Figura 2a)

- *Prefija*: En esta notación el operador precede a los operandos. El orden de operaciones queda claramente especificado por la posición de los elementos de la expresión (operadores y operandos). Se puede expresar como una lista de elementos o separando los operandos de cada operación con comas y entre paréntesis.

- *Infija*: Se trata de la notación que utilizamos habitualmente. El operador se sitúa entre sus operandos (en el caso de los operadores binarios). En el caso de operadores unarios (como el coseno) cómo solo hay un operador se usa la misma forma que en la notación prefija.

- *Postfija*: Exactamente igual que en el caso de la notación prefija, aunque en este caso el operador se sitúa detrás de los operandos. De hecho, para pasar una expresión de prefija a postfija lo único que tenemos que hacer es invertir la lista de elementos que componen la expresión.

En nuestro caso elegimos la forma postfija con paréntesis y comas. El motivo de esta elección reside en que la información se debe almacenar en una cadena de caracteres y que de esta forma resultará más fácil indicar la máxima información sobre posibles errores sintácticos que pueda contener la cadena de caracteres que queremos analizar. Además, restringiremos los operadores a unarios y binarios (ya que prácticamente todos los operadores lo son). (Figura 2b)

CONSTRUCCIÓN DEL ÁRBOL EXPRESIÓN

Hasta ahora el resultado de todas las transformaciones que hemos realizado han sido cadenas de caracteres. El siguiente paso será obtener lo que se denomina el árbol expresión. Consiste en una estructura de datos con forma de árbol donde las hojas son operandos y el resto de hojas son operadores cuyos operandos son sus hijos. (Figura 3a)

El procedimiento para la generación del árbol expresión consiste en diferenciar en la cadena de caracteres el operador de mayor prioridad (que constituirá el nodo raíz del árbol expresión) y cada uno de sus operadores, (cosa que en nuestro caso será casi inmediato ya que hemos reescrito la expresión a notación postfija). Luego, para cada operador, repetiremos el proceso hasta que los operadores sean expresiones atómicas, entendiendo este tipo de expresiones como todo aquello que sea una constante o el identificador de una variable.

El árbol expresión es una estructura de datos en forma de árbol

Como puede observarse se trata de un procedimiento recursivo, ya que cada operando de una expresión puede a su vez ser una subexpresión con un operador y uno o varios operandos, y así sucesivamente.

Notación	Ejemplo
Prefija (lista)	* + a b ^ c d
Prefija	*(+(a,b),^(c,d))
Infija	(a+b)*c^d
Postfija	((c,d)^,(a,b)+)*

Figura 2a. Diferentes notaciones matemáticas.

((a+b)*(c^d)) ⇒ ((c,d)^,(a,b)+)*

Figura 2b. Pasar a notación postfija.

Existen, básicamente dos variantes de dicho procedimiento dependiendo de en qué orden se hagan las llamadas recursivas. La primera se denomina *width first* (primero en anchura) y consiste en generar el árbol expresión por niveles de profundidad. Primero generamos el nodo raíz, luego sus hijos, luego los hijos de sus hijos, etc. (Figura 3b) En este caso lo preferible es tener la cadena de caracteres en notación postfija. La segunda variante se denomina *depth first* (primero profundidad), consiste en generar el árbol de la siguiente manera: para cada nodo correspondiente a un operador binario analizaremos primero todo su subárbol derecho y después todo su subárbol izquierdo, y así sucesivamente. (Figura 3c) En este caso lo preferible es tener la cadena de caracteres en notación prefija. La ventaja de este método sobre el anterior es que la cadena de caracteres sólo se lee una vez. La desventaja es que con el primer método es más fácil especificar qué errores sintácticos contiene la expresión, en el caso de que los haya, para poder informar al usuario.

El algoritmo recursivo para generar el árbol expresión partiendo de la cadena expresión en notación postfija usando la variante *width first* puede concretarse en los siguientes pasos:

Algoritmo analizar_expresión:

- Paso 1. Llamaremos a **analizar_subexpresión** pasándole como pará-

metro la cadena introducida por el usuario.

- Paso 2. Devolver el puntero correspondiente al nodo raíz del árbol expresión. Terminar.

Algoritmo analizar_subexpresión:

- Paso 1. Leeremos el primer carácter. Si se trata de un '(' la subexpresión no es atómica. Es decir, tiene al menos un operador. Saltaremos al paso 2. En caso contrario leeremos toda la cadena y comprobaremos que corresponda a una constante numérica o al nombre de una variable. Crearemos un nodo hoja de la estructura árbol. En él almacenaremos el valor de la constante o el identificador de la variable (según corresponda) y devolveremos un puntero que apunte a dicho nodo. Terminar.

- Paso 2. Leeremos contando el número de '(' y de ')' hasta que los dos contadores valgan lo mismo y el siguiente carácter sea un ')' o una ','. Lo que hemos leído hasta ahora es el primer operando. Llamaremos a la función **analizar_subexpresión** pasando como parámetro la subcadena que hemos leído.

Width first crea el árbol de expresión por niveles de profundidad

- Paso 3. Si el siguiente carácter era un ')' se trata de un operador unario. Saltar al Paso 6.

- Paso 4. Si por el contrario era una ',' se trata de un operador binario. Seguir.

- Paso 5. Seguiremos leyendo contando el número de '(' y de ')' hasta que los dos contadores valgan lo mismo y el siguiente carácter sea un ')'. Lo que hemos leído hasta ahora es el segundo operando. Llamaremos a la función **analizar_subexpresión** pasando como parámetro la subcadena correspondiente al segundo operando.

- Paso 6. Seguiremos leyendo hasta el final de la cadena. Lo que leemos es el operador. Comprobaremos que se trata de un operador válido y que hemos leído el número adecuado de operandos. Crearemos un nodo de la estructura árbol expresión. Haremos que los apuntadores de dicho nodo apunten a los subárboles resultantes de analizar los operandos y devolveremos un puntero que apunte al nodo del operador. Terminar.

El objetivo de haber construido el árbol de expresión consiste en dar valores a las variables

Si en cualquier momento de este proceso nos encontramos con el final de la cadena de caracteres, o con un carácter inesperado, retornaremos un código de error correspondiente y eliminaremos el árbol generado.

CALCULO DE VALORES DE LA FUNCIÓN

El objetivo de haber construido el árbol expresión es el de poder dar valores a las variables y evaluar la expresión para ver que el número que devuelve. Para ello usaremos un algoritmo que recorrerá el

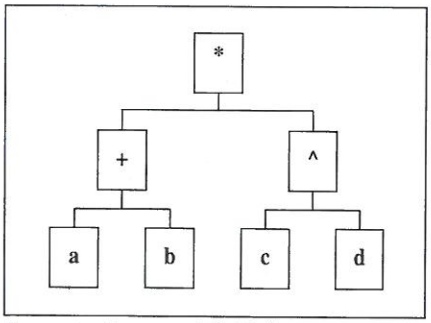


Figura 3a. Esquema del árbol expresión.

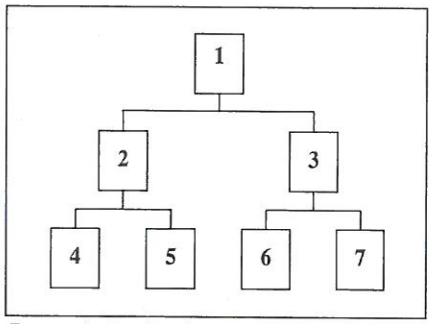


Figura 3b. Cuadro de operandos y operadores.

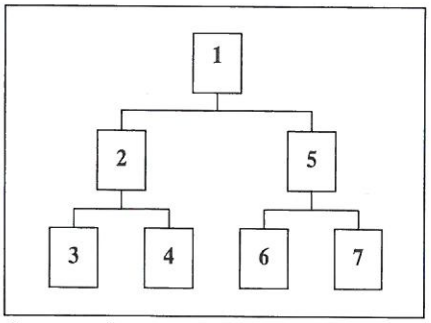


Figura 3c. Caracteres en notación prefija.

árbol en el mismo orden que el anterior lo generó.

Algoritmo calcular_expresión:

- Paso 1. Para cada nodo hoja correspondiente a una variable le asignaremos su valor actual.

- Paso 2. Llamar a **calcular_subexpresión** pasándole como parámetro el nodo raíz del árbol expresión.

- Paso 3. El valor que devuelve la función es precisamente el valor de la expresión. Retornar dicho valor y terminar.

Algoritmo calcular_subexpresión:

- Paso 1. Comprobaremos si el nodo pasado como parámetro es un nodo hoja. De ser así, retornaremos el valor que tenga asignado y terminaremos.
- Paso 2. En caso contrario estamos delante de un operador. Para cada uno de sus operandos (que estarán apuntados desde este nodo) llamamos a la función calcular_subexpresión pasando el operando como parámetro.
- Paso 3. Una vez que las llamadas recursivas devuelvan los valores de evaluar las subexpresiones correspondientes a los operandos, se operarán entre sí mediante el operador del nodo actual. Devolveremos el valor resultante y terminaremos.

Las variables válidas serán la X y la Y

Se podrían integrar los algoritmos (*analizar_subexpresión* y *calcular_subexpresión*) en uno, pero en nuestro caso no sería conveniente, ya que lo que queremos es generar una vez el árbol expresión y luego calcular muchos valores de dicha expresión sin tener que volver a analizarla. Como en nuestro caso las expresiones corresponden a funciones tridimensionales, las variables válidas serán la **X** y la **Y**, y el resultado de evaluar la expresión será la **Z**. Teniendo en cuenta que sólo hay dos variables válidas realizaremos la siguiente modificación del algoritmo *analizar_subexpresión* que hemos visto en el apartado anterior: tendremos dos nodos hoja, (uno para la *X* y otro para la *Y*) a donde apuntaran todos los operadores que los tengan como parámetros. De esta manera variando el campo donde almacenamos el valor de la *X* y la *Y* y ejecutando el algoritmo *calcular_subexpresión* iremos obteniendo los diferentes valores de la función que se va a representar. Esta modificación también puede resultar útil en el caso genérico.

Tabla 1. Cálculo de valores de la función.

- FUNCTION TECLAS: Lee una tecla del teclado.
- PROCEDURE FINESTRA: Dibuja una ventana con un gradiente determinado.
- PROCEDURE NUM_OUT: Imprime por pantalla un número.
- FUNCTION NUM_IN: Obtiene un número procedente del teclado.
- PROCEDURE MANEL: Dibuja la 'portada'.
- PROCEDURE CrearExpresión: Genera el árbol expresión.
- FUNCTION CALCULA_SUBEXPRESIO: Función recursiva para calcular el valor de un subárbol expresión.
- FUNCTION CALCULA_EXPRESIO: Función que calcula el valor del árbol expresión para una 'X' y una 'Y' dadas.
- PROCEDURE CALCULAR_COLOR: Determina a qué color del arco iris corresponde un punto de la función
- PROCEDURE CALCULAR_Z: Si no está generado el árbol expresión llama a 'CALCULA_EXPRESIO'.
- PROCEDURE INTRODUIR_CANTONADES: Cada vez que se cambia el dominio de la función recalcula los valores de las esquinas de la matriz de valores.
- PROCEDURE CALCULAR_MARGE: Calcula la proyección XY de un margen de la matriz de valores.
- PROCEDURE CALCULAR_MATRIU: Calcula la proyección XY de la matriz de valores.
- PROCEDURE CALCULAR: Controla mediante 'flags' que parámetros deben recalcularse.
- PROCEDURE QUI_PRIMER: Mira por que esquina debe empezar el algoritmo del pintor.
- PROCEDURE PRIMER: Traza el primer margen de la matriz.
- PROCEDURE DIBUIXAR_QUADRAT(P:BOOLEAN): Dibuja la proyección XY de los márgenes de la función.
- PROCEDURE DIBUIXAR_MATRIU: Traza toda la matriz.
- PROCEDURE PINTAR_TRIANGLE: Pinta con el gradiente adecuado una cara de la matriz.
- PROCEDURE PINTAR_DIBUIX: Pinta toda la matriz.
- PROCEDURE CANVIAR_FUNCIO: Permite al usuario cambiar la función
- PROCEDURE CANVIAR_DOMINI: Permite cambiar el dominio 'X' e 'Y' de la área a graficar.
- PROCEDURE APUNTA_FOCUS: Subrutina usada por 'CANVIAR_FOCUS'.
- PROCEDURE CANVIAR_FOCUS: Permite cambiar la orientación del foco de luz.
- PROCEDURE CANVIAR_PERSPECT: Permite cambiar la orientación de la vista determinando los ángulos de caída y deriva.
- FUNCTION PROPOR_Z: Para cambiar la proporción del eje 'Z' respecto los ejes 'X' e 'Y'.
- FUNCTION CONFIGCOLOR: Para configurar que colores se usarán.
- PROCEDURE DADES_INICIALS: Reasigna a todas las variables sus valores por defecto.
- PROCEDURE FI: Finaliza la ejecución del programa.
- CUERPO PRINCIPAL: Donde se muestra el menú principal.

PROYECCIÓN DE LA FUNCIÓN SOBRE EL PLANO

Llegados a este punto, ya somos capaces de analizar cualquier expresión matemática correspondiente a una función tridimensional introducida mediante el teclado por el usuario. También conocemos la forma de calcular valores de dicha función. El siguiente paso consistirá en representar dicha tabla de valores en pantalla.

Para ello lo primero que hemos de considerar es el problema de la proyección de la función tridimensional sobre la pantalla (que sólo tiene dos dimensiones).

Existen básicamente dos alternativas: usar la perspectiva **cónico-frontal** o usar una proyección **plana**. La primera tiene como ventaja que ofrece un efecto mucho más realista. Por el contrario, la segunda, al ser una vista en lugar de una perspectiva no distorsiona las proporciones. Como lo que queremos representar es una función matemática escogeremos la segunda alternativa.

La perspectiva cónico-frontal, ofrece un efecto bastante más realista

Para pasar de las tres dimensiones a dos lo hacemos mediante un ángulo de caída y otro de deriva que el usuario podrá determinar. (Figura 4a)

Para ver cómo se programaría la proyección en perspectiva y cómo realizar los giros de caída y deriva, deberán consultar otros números de la revista donde se haya tratado el tema.

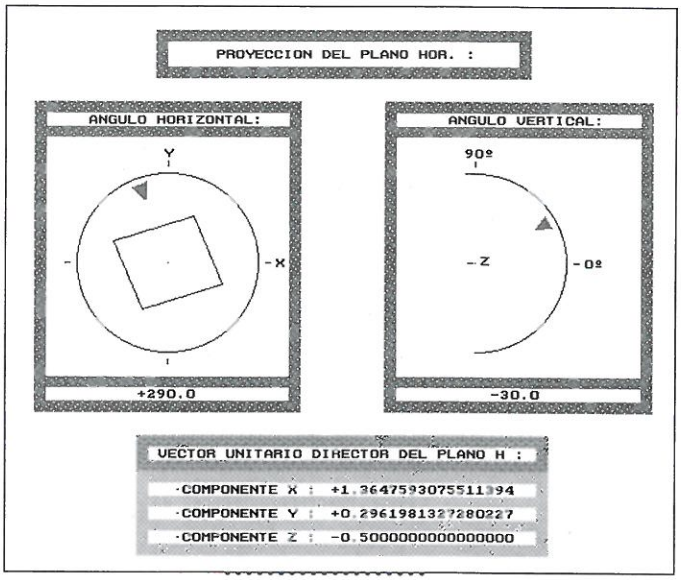


Figura 4a. El programador puede determinar el ángulo de caída y el de deriva.

REPRESENTACIÓN GRÁFICA EN "ALÁMBRICO"

Una vez que ya sabemos proyectar los puntos de la función sobre la pantalla podemos hacer una primera representación gráfica de la matriz de valores. Para ello tan sólo tendremos que decidir los intervalos de X e Y que configurarán el dominio de la función que vamos a representar. A continuación, se generará la matriz de valores y luego se trazarán líneas entre las coordenadas de pantalla de los puntos que sean adyacentes en la matriz de valores. Hemos decidido trazar una línea adicional 'la diagonal' para obtener 'caras' triangulares, ya que nos ira bien en apartados posteriores.

Si bien se trata de una representación un tanto tosca, proporciona una imagen muy exacta de que apariencia que tiene la función. (Figura 5a) Cambiando el dominio y los ángulos de la proyección podremos ir obteniendo diferentes vistas de la función.

Aunque, por supuesto, no vamos a contentarnos con representaciones alám-

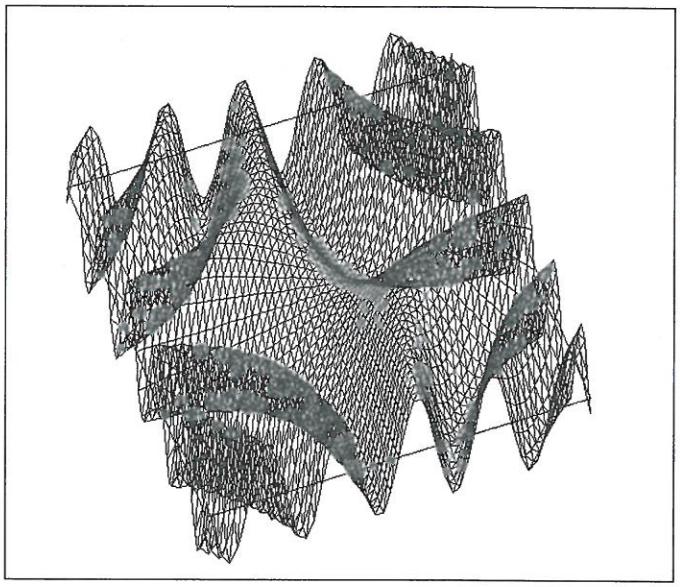


Figura 5a. Obtendremos diferentes perspectivas variando el dominio y los ángulos de proyección.

$$v_x = B_x - A_x$$
$$v_y = B_y - A_y$$
$$v_z = B_z - A_z$$

$$u_x = C_x - A_x$$
$$u_y = C_y - A_y$$
$$u_z = C_z - A_z$$

$$\vec{vd} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ v_x & v_y & v_z \\ u_x & u_y & u_z \end{vmatrix}$$

$$vd_x = v_y u_z - v_z u_y$$
$$vd_y = v_z u_x - v_x u_z$$
$$vd_z = v_x u_y - v_y u_x$$

$$Cos(\alpha) = -\frac{vd_x\, vf_x + vd_y\, vf_y + vd_z\, vf_z}{\sqrt{vd_x^{\,2} + vd_y^{\,2} + vd_z^{\,2}}}$$

$$Porcentaje_luz = 100Cos(\alpha)$$

Figura 6b. Fórmulas usadas para calcular el porcentaje de iluminación de una cara formada por los puntos A, B y C (en orden levogiro) con el vector característico del foco impropio (vf).

bricas, en los siguientes apartados aprenderemos cómo incorporar un foco de luz para poder generar sombreados que den a las representaciones gráficas el efecto de relieve sólido.

ILUMINACIÓN Y FOCOS DE LUZ

Para poder iluminar una será necesario incorporar luz. Los objetos que emiten luz se denominan focos de luz, existiendo una gran variedad de ellos. Una de las muchas clasificaciones de focos de luz los divide en propios e impropios. Los primeros son aquellos que se encuentran a distancia finita del objeto que iluminar y radian en todas las direcciones. Los impropios son aquellos que se encuentran a una distancia infinita y por tanto se comportan como si sólo radiaran en una dirección. Como ejemplo intuitivo se puede pensar en los focos propios como si fueran bombillas y en los impropios como si fueran el sol. Por tanto los focos de luz propios vendrán caracterizados por una posición en el espacio y los impropios por un vector.

Los focos propios son los que se encuentran a distancia finita del objeto que iluminan

Como estamos iluminando una función matemática, dicha iluminación debe servir para darnos más información sobre la orientación y la pendiente de sus diferentes zonas. Es por esto que elegimos un foco impropio en nuestro caso, ya que de este modo iluminará de forma uniforme toda la superficie de la representación.

En nuestro caso concreto lo más intuitivo será permitir que el usuario pueda modificar la orientación de su vector característico desde el programa mediante los siguientes parámetros: ángulo sobre la superficie plana y ángulo de elevación. (Figura 6a)

Con esta caracterización del vector del foco y la posición de los tres puntos de la matriz de valores que conforman cada una de las caras podemos calcular que porcentaje de iluminación incide sobre ellas. (Figura 6b)

EL USO DE GRADIENTES

Una vez ya sabemos qué porcentaje de luz ilumina cada cara podemos pintarla del color que le corresponda: es decir el color del foco reduciendo el parámetro de luminosidad en el porcentaje de luz incidente. Pero, ¿qué ocurre si la resolución en la que trabajamos sólo soporta un reducido número de colores simultáneos en pantalla?, por ejemplo, en una resolución con 16 colores. En ese caso no tendremos más remedio que recurrir al uso de los gradientes.

Un gradiente es una textura o patrón de *pixels* de dos o más colores que intenta simular el color que tiene como componentes RGB la suma ponderada de las componentes RGB de los colores que lo componen. En nuestro caso usaremos gradientes bicolores, donde un color será el color del foco de luz y el otro será el negro (la oscuridad). El porcentaje de luz

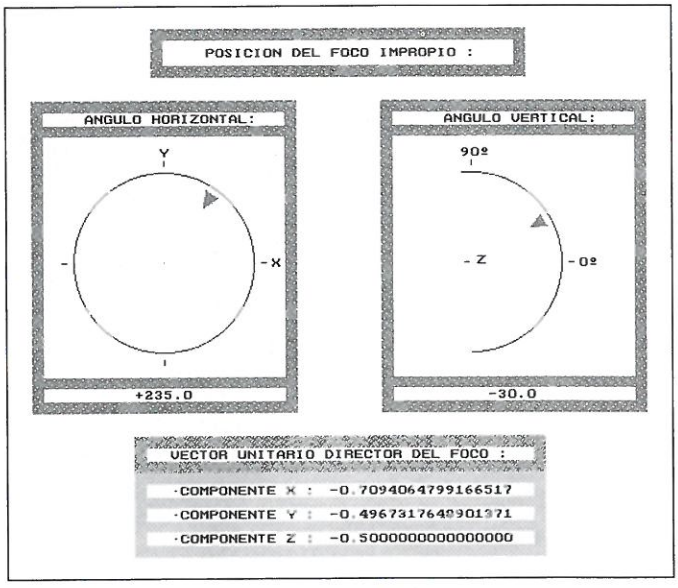


Figura 6a. Cambiaremos la orientación del vector característico mediante los valores de los ángulos.

que simulará el gradiente será precisamente el porcentaje de *pixels* del color del foco de luz respecto del número total de *pixels*.

Para aprovechar ciertas subrutinas, que podemos encontrar en las librerías gráficas de buena parte de los compiladores de lenguaje de alto nivel, usaremos gradientes de 8x8 *pixels*. Cada gradiente tendrá dos *pixels* coloreados de diferencia con sus adyacentes. En concreto, como los gradientes serán de 64 *pixels*, usaremos 33 gradientes.

Los gradientes se confeccionarán de tal forma que se provoquen los mínimos efectos laterales posibles. (Ver los gradientes del programa de ejemplo). Los efectos laterales se producen cuando el gradiente no está bien distribuido y al poner varios patrones, uno al lado de otro, se observan como 'cenefas' de pixeles.

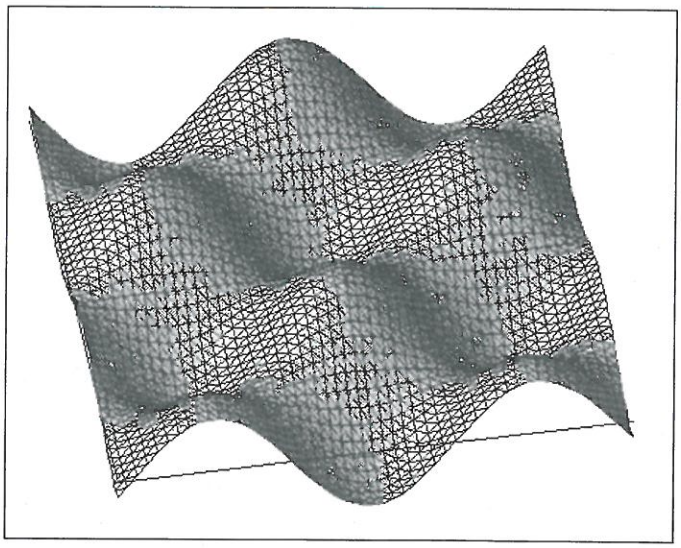


Figura 7a1. Resultado final de la aplicación.

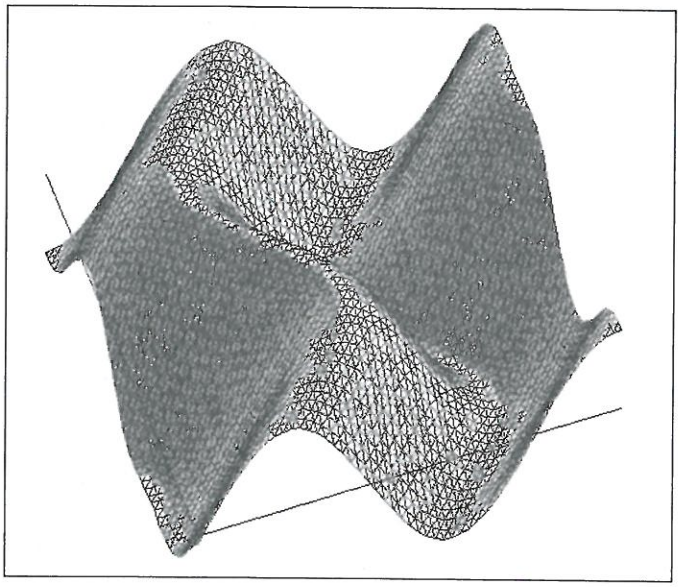


Figura 7a2. Otro posible resultado de la misma aplicación.

REPRESENTACIÓN GRÁFICA EN "SÓLIDO"

Llegados a este punto ya sabemos calcular, para cada cara de la matriz de valores, el porcentaje de luz que le proporciona el foco impropio. Y por tanto, con qué gradiente hay que pintar cada cara. Podemos pues, proceder a la representación gráfica en "sólido". El algoritmo será prácticamente el mismo que en la representación en alámbrico, aunque hay un detalle que se deberá tener en cuenta, es muy probable que alguna cara se solape total o parcialmente con otra. Para evitarlo podemos usar el conocido algoritmo del pintor. Lo cierto es que este algoritmo es muy poco adecuado para la mayoría de las representaciones gráficas de mundos tridimensionales. En mundos donde hay mucho número de caras es costoso ordenarlas y pintar muchas caras que al final quedarán ocultas. Sin embargo nuestro caso es diferente, puesto que ya conocemos el orden en que hay que pintar las caras. Además,

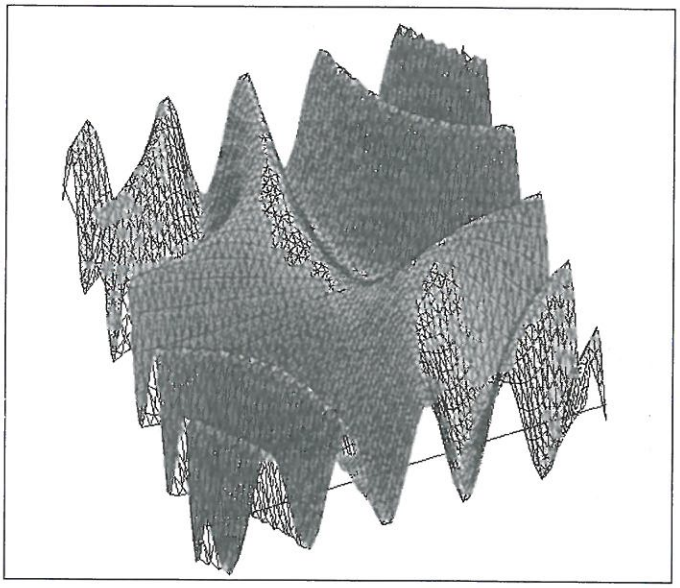


Figura 7a3. Tercer resultado de la aplicación.

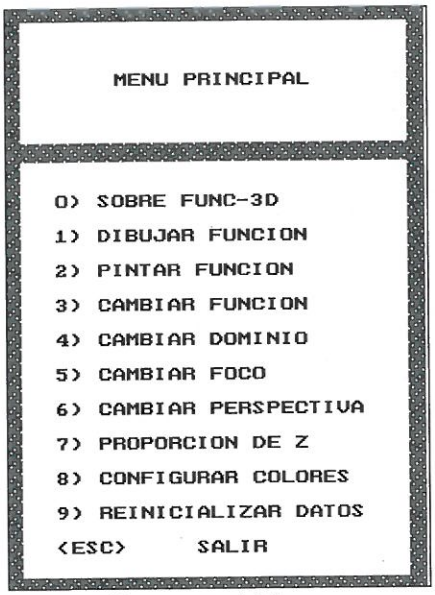


Figura 8a. Menú principal del programa.

no habrá muchas caras que queden ocultas del todo y no tenemos demasiadas caras que pintar.

El orden en que hay que pintar las caras es el siguiente: se empieza por la esquina más lejana de la matriz de valores y se recorre la matriz hacia su segunda esquina más lejana. Se va repitiendo el proceso hasta que se acaban las filas de la matriz.

(En la Figura 7a podemos observar el resultado final)

EL PROGRAMA DE EJEMPLO

El artículo viene acompañado de un programa de ejemplo (llamado func-3d) para que no se quede todo en conceptos teóricos. Dicho programa permite al usuario introducir la función que representar en notación postfija (con paréntesis y comas). Una vez efectuado este paso tendremos que especificar el dominio que queremos ver, la orientación del foco de luz, la perspectiva, la proporción de eje Z respecto del X y el Y; y configurar los colores. (De hecho todas las capturas de pantalla que aparecen en el artículo corresponden al programa en cuestión).

Los operandos permitidos son: números, **X**, **Y**, **Pi** y **e**.

Y los operadores permitidos son: + (suma), - (resta), * (multiplicación), / (división), **Abs** (valor absoluto), **ArcTan** (arcotangente), **Cos** (coseno), **Ln** (logaritmo neperiano), **Sin** (seno), **Sqrt** (raíz cuadrada) y **Tan** (tangente).

Además se incluye, en la representación en alámbrico, la posibilidad de que el color de los trazos dependa del valor de la componente *Z* (usando los colores del arco iris).

El código del programa está escrito en Pascal con lo cual ningún programador experimentado tendrá problemas para traducirlo a su lenguaje preferido. El programa se puede compilar sin ningún problema en el compilador de Borland 'Turbo Pascal' (versión 6 o superior). No obstante se acompaña también el fichero ejecutable (que deberá copiarse junto con el fichero 'egavga.bgi' y los ficheros con extensión '*.chr' para que funcione).

Func-3d permite representar la función en notación postfija

Los requisitos de sistema para ejecutarse son mínimos: Funcionará en cualquier ordenador (286 o superior) con DOS.

Como siempre, se anima al lector a que explore el código fuente y haga modificaciones para terminar de comprender los conceptos que en el artículo se explican.

Los módulos del programa

Todo el código fuente del programa está en un único fichero que incorpora los siguientes procedimientos y funciones.

Función	Input del programa
10*Sin(Sin(X)+Y))	((((Y)Sin,X)+)Sin,10)*
Sin(X)*Cos(Y)	((X)Sin,(Y)Cos)*
-Sin(X*Y)	(((X,Y)*)Sin),-1)*

Figura 8b. Sugerencias para el programa.

Todo aquel que desee analizar el código correspondiente al programa de representación gráfica de funciones, podrá encontrarlo en el CD-ROM de la revista.

Ejecutando el programa

Para probar el programa copiaremos el fichero ejecutable func-3d.exe, el fichero egavga.bgi y los ficheros con extensión chr en un mismo directorio y ejecutaremos. aparecerá una portada de presentación. Pulsaremos cualquier tecla y pasaremos al menú principal del programa. (Figura 8a) Una vez allí podemos pulsar la tecla 1 y ver la representación en alambrico de la función por defecto (-Sin(X*Y)); o pulsar 2 y ver la representación en sólido. Luego si queremos podemos modificar la orientación del foco de luz pulsando el 5 o los ángulos de caída y deriva de la proyección pulsando 6.

Pulsando el número **3** cambiamos la función, (en la Figura 8b vemos algunas sugerencias) y pulsando el **4** cambiamos de dominio.

Cuando cambiemos la función o el dominio a representar tendremos que tener cuidado de elegir una proporción de *Z* adecuada (opción **7** del menú principal), ya que si la función obtiene valores muy extremos la representación se saldrá de la pantalla. (Sin duda, un punto que mejorar en el programa)

La opción **8** (configurar colores) sirve para elegir con que color se traza la función, los márgenes, etc.

Si nos liamos mucho y queremos volver a los valores por defecto, bastará con pulsar la tecla **9**. Y para salir basta con pulsar la tecla de escape en el menú principal.

Programación del módem (y IV)

Constantino Sánchez Ballesteros (constantino@redestb.es)

INTRODUCCIÓN

Este es el último artículo de la serie programación del módem, en el que se abordará la finalización del proyecto TERMINAL. En los tres anteriores hemos ido desmenuzando cada uno de los formularios y módulos que componen el proyecto: **CANSEND.frm**, **frmPROPS.frm** y **VBTERM.glo**. Como vimos en el artículo anterior, habíamos dejado incompleto el detalle del formulario principal **VBTERM.frm**, el cual se encarga de llevar a buen fin todas las instrucciones dirigidas al programa mediante las opciones del menú que hemos agregado al formulario. Comenzamos en el punto de código donde acabamos el mes pasado para concluir el desarrollo de esta última parte.

CONTINUACIÓN DE VBTERM.FRM

El último procedimiento que vimos de este formulario era **Private Sub mnuHangup_Click()**. Así pues, continuamos con el siguiente:

```
' Muestra el valor de la propiedad CDHolding Private Sub mnuHCD_Click()
  If MSComm1.CDHolding Then
    Temp = "True"
  Else
    Temp = "False"
  End If
  MsgBox "CDHolding = " + Temp
End Sub

' Muestra el valor de la propiedad CTSHolding
Private Sub mnuHCTS_Click()
  If MSComm1.CTSHolding Then
    Temp = "True"
  Else
    Temp = "False"
  End If
  MsgBox "CTSHolding = " + Temp
End Sub

' Muestra el valor de la propiedad DSRHolding (
Private Sub mnuHDSR_Click()
  If MSComm1.DSRHolding Then
    Temp = "True"
  Else
    Temp = "False"
  End If
  MsgBox "DSRHolding = " + Temp
End Sub
```

El siguiente procedimiento establece la propiedad *InputLen*, que determina el número de bytes de datos leídos cada vez que se usa *Input* para recuperar datos del buffer de entrada. Al establecer **0** en *InputLen* se especifica que debe leerse todo el contenido del buffer.

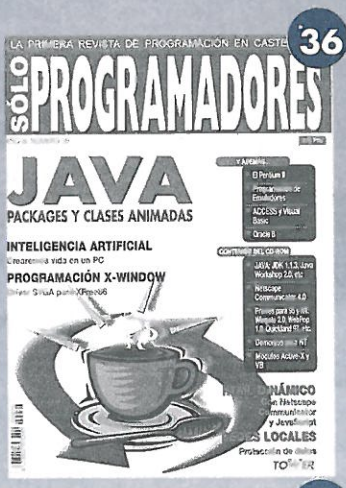
```
Private Sub mnuInputLen_Click()
  On Error Resume Next
```

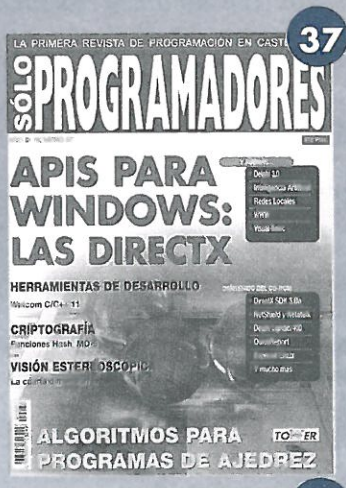
En este artículo continuamos con el desarrollo del programa Terminal retomando el formulario principal del proyecto.
Con esta cuarta entrega daremos por finalizada la programación del módem una vez vistos todos los formularios y módulos del del proyecto y completando el formulario principal.

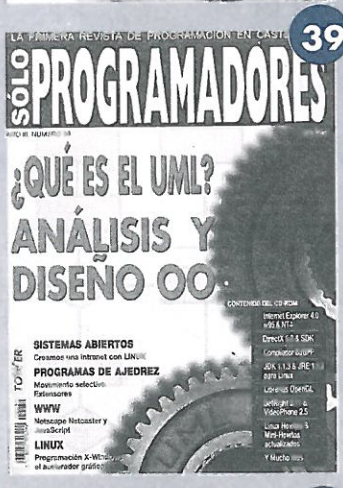
números atrasados

SÓLO PROGRAMADORES

completa ya tu colección







suscríbete

a Sólo Programadores y consigue un magnífico descuento

suscripción normal

ahorro **20%**

12 revistas (1 año) por sólo...

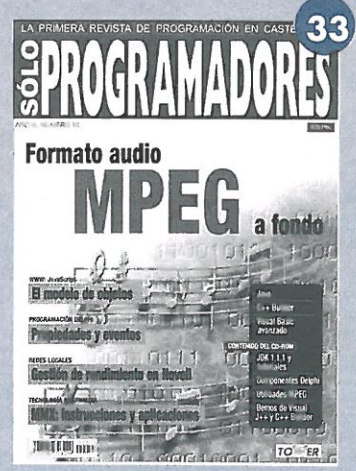
9.350 ptas.

suscripción estudiantes (carreras técnicas)

ahorro **40%**

12 revistas (1 año) por sólo...

7.050 ptas.



```
    Temp = InputBox$("Introduzca nuevo
        InputLen:", "InputLen",
        Str$(MSComm1.InputLen))
    If Len(Temp) Then
      MSComm1.InputLen = Val(Temp)
      If Err Then MsgBox Error$, 48
    End If
End Sub

    ' Muestra el formulario de propiedades de CommPort
Private Sub mnuProperties_Click()
    frmProperties.Show vbModal
End Sub

    ' Alterna el estado del puerto (abierto o cerrado)
Private Sub mnuOpen_Click()
    On Error Resume Next
    Dim OpenFlag

    MSComm1.PortOpen = Not
        MSComm1.PortOpen
    If Err Then MsgBox Error$, 48

    OpenFlag = MSComm1.PortOpen

    mnuOpen.Checked = OpenFlag
    mnuSendText.Enabled = OpenFlag
    tbrToolBar.Buttons("TransmitTextFile").Enabled
        = OpenFlag

    If MSComm1.PortOpen Then
        ' Habilita el botón de marcar y el elemento de menú asociado
        mnuDial.Enabled = True

        tbrToolBar.Buttons("DialPhoneNumber").En
        abled = True

        ' Habilita el botón de colgar y el elemento de menú asociado
        mnuHangUp.Enabled = True
        tbrToolBar.Buttons("HangUpPhone").Enabled
        = True

        imgConnected.ZOrder
        sbrStatus.Panels("Settings").Text =
        "Valores:" & MSComm1.Settings
        StartTiming
    Else

        ' Habilita el botón de marcar y el elemento de menú asociado
        mnuDial.Enabled = True
        tbrToolBar.Buttons("DialPhoneNumber").Enabled
        = True

        ' Deshabilita el botón de colgar y el elemento de menú asociado
        mnuHangUp.Enabled = False
        tbrToolBar.Buttons("HangUpPhone").Enabled
        = False

        imgNotConnected.ZOrder
        sbrStatus.Panels("Settings").Text =
        "Valores:"
        StopTiming
    End If
End Sub

    ' Abre el archivo de registro
Private Sub mnuOpenLog_Click()
    Dim replace
    On Error Resume Next
    OpenLog.Flags = cdlOFNHideReadOnly Or
        cdlOFNExplorer
    OpenLog.CancelError = True

    ' Obtiene del usuario el nombre de archivo largo
    OpenLog.DialogTitle = "Abrir archivo de registro de comunicaciones"
    OpenLog.Filter = "Archivos de registro
        (*.LOG)|*.log|Todos los archivos (*.*)|*.*"

    Do
      OpenLog.filename = ""
      OpenLog.ShowOpen
      If Err = cdlCancel Then Exit Sub
      Temp = OpenLog.filename

      ' Si el archivo ya existe, pregunta al usuario si desea sobrescribirlo o agregarle datos
      Ret = Len(Dir$(Temp))
      If Err Then
        MsgBox Error$, 48
        Exit Sub
      End If
      If Ret Then
        replace = MsgBox("Reemplazar el archivo -
        " + Temp + "?", 35)
      Else
        replace = 0
      End If
    Loop While replace = 2

    ' El usuario ha hecho clic en el botón Sí, así que elimina el archivo
    If replace = 6 Then
      Kill Temp
      If Err Then
        MsgBox Error$, 48
        Exit Sub
      End If
    End If

    ' Abre el archivo de registro
    hLogFile = FreeFile
    Open Temp For Binary Access Write As
        hLogFile
    If Err Then
      MsgBox Error$, 48
      Close hLogFile
      hLogFile = 0
      Exit Sub
    Else
      ' Va al final del archivo para poder agregar nuevos datos
      Seek hLogFile, LOF(hLogFile) + 1
    End If

    frmTerminal.Caption = "Terminal de Visual
        Basic - " + OpenLog.FileTitle
    mnuOpenLog.Enabled = False
    tbrToolBar.Buttons("OpenLogFile").Enabled =
        False
    mnuCloseLog.Enabled = True
    tbrToolBar.Buttons("CloseLogFile").Enabled =
        True
End Sub
```

El siguiente procedimiento establece la propiedad *ParityReplace*, que contiene el carácter que reemplazará a los caracteres incorrectos recibidos a causa de un error de paridad.

```
Private Sub mnuParRep_Click()
    On Error Resume Next

    Temp = InputBox$("Escriba el carácter a reemplazar", "ParityReplace",
        frmTerminal.MSComm1.ParityReplace)
    frmTerminal.MSComm1.ParityReplace =
        Left$(Temp, 1)
    If Err Then MsgBox Error$, 48
End Sub
```

Este procedimiento establece la propiedad *Rthreshlold* que determina el número de bytes que pueden llegar al buffer de recepción antes de disparar el evento *OnComm*. Además, se ocupa de

que se establezca **comEvReceive** en la propiedad *CommEvent*.

```
Private Sub mnuRThreshold_Click()
  On Error Resume Next

  Temp = InputBox$("Introduzca el nuevo
      RThreshold:", "RThreshold",
      Str$(MSComm1.RThreshold))
  If Len(Temp) Then
    MSComm1.RThreshold = Val(Temp)
    If Err Then MsgBox Error$, 48
  End If

End Sub
```

```
' El evento OnComm se usa para interceptar eventos y errores de comunicaciones
Private Static Sub MSComm1_OnComm()
  Dim EVMsg$
  Dim ERMsg$

' Bifurca (desvía) según la propiedad CommEvent
  Select Case MSComm1.CommEvent
    ' Mensajes de evento
    Case comEvReceive
      Dim Buffer As Variant
      Buffer = MSComm1.Input
      Debug.Print "Recibir - " & StrConv(Buffer,
      vbUnicode)
      ShowData txtTerm, (StrConv(Buffer,
      vbUnicode))
    Case comEvSend
    Case comEvCTS
      EVMsg$ = "Detectado cambio en CTS"
    Case comEvDSR
      EVMsg$ = "Detectado cambio en DSR"
    Case comEvCD
      EVMsg$ = "Detectado cambio en CD"
    Case comEvRing
      EVMsg$ = "El teléfono está sonando"
    Case comEvEOF
      EVMsg$ = "Detectado el final del archivo"

    ' Mensajes de error
    Case comBreak
      ERMsg$ = "Parada recibida"
    Case comCDTO
      ERMsg$ = "Sobrepasado el tiempo de
      espera de detección de portadora"
    Case comCTSTO
      ERMsg$ = "Soprepasado el tiempo de
      espera de CTS"
    Case comDCB
      ERMsg$ = "Error recibiendo DCB"
    Case comDSRTO
      ERMsg$ = "Sobrepasado el tiempo de
      espera de DSR"
    Case comFrame
      ERMsg$ = "Error de marco"
    Case comOverrun
      ERMsg$ = "Error de sobrecarga"
    Case comRxOver
      ERMsg$ = "Desbordamiento en el búfer
      de recepción"
    Case comRxParity
      ERMsg$ = "Error de paridad"
    Case comTxFull
      ERMsg$ = "Búfer de transmisión lleno"
    Case Else
      ERMsg$ = "Error o evento desconocido"
  End Select

  If Len(EVMsg$) Then
    ' Muestra los mensajes de evento en la barra de estado
    sbrStatus.Panels("Status").Text = "Estado:"
    & EVMsg$

    ' Activa el cronómetro para que el mensaje de la barra de estado se borre después de dos segundos
    Timer2.Enabled = True

  ElseIf Len(ERMsg$) Then
    ' Muestra los mensajes de evento en la barra de estado
    sbrStatus.Panels("Status").Text = "Estado:"
    & ERMsg$

    ' Muestra los mensajes de error en un cuadro de alerta
    Beep
    Ret = MsgBox(ERMsg$, 1, "Haga clic en
    Cancelar para salir, clic en Aceptar para
    ignorar.")

    ' Si el usuario hace clic en Cancelar (2)...
    If Ret = 2 Then
      MSComm1.PortOpen = False ' Cierra el puerto y sale.
    End If

    ' Activa el cronómetro para que el mensaje de la barra de estado se borre después de dos segundos
    Timer2.Enabled = True
  End If
End Sub
```

Utilizamos la opción de menú **Transmitir archivo de texto** para enviar datos al receptor del mensaje:

```
Private Sub mnuSendText_Click()
  Dim hSend, BSize, LF&

  On Error Resume Next

  mnuSendText.Enabled = False
  tbrToolBar.Buttons("TransmitTextFile").Enabled
      = False

  ' Obtiene del usuario el nombre del archivo de texto
  OpenLog.DialogTitle = "Enviar archivo de
      texto"
  OpenLog.Filter = "Archivos de texto
      (*.TXT)|*.txt|Todos los archivos (*.*)|*.*"
  Do
    OpenLog.CancelError = True
    OpenLog.filename = ""
    OpenLog.ShowOpen
    If Err = cdlCancel Then
      mnuSendText.Enabled = True

      tbrToolBar.Buttons("TransmitTextFile").Ena
      bled = True
      Exit Sub
    End If
    Temp = OpenLog.filename

    ' Si el archivo no existe, vuelve
    Ret = Len(Dir$(Temp))
    If Err Then
      MsgBox Error$, 48
      mnuSendText.Enabled = True

      tbrToolBar.Buttons("TransmitTextFile").Ena
      bled = True
      Exit Sub
    End If
    If Ret Then
      Exit Do
    Else
      MsgBox Temp + " no encontrado.", 48
    End If
  Loop

  ' Abre el archivo de registro
  hSend = FreeFile
  Open Temp For Binary Access Read As hSend
```

```
If Err Then
  MsgBox Error$, 48
Else
    ' Muestra el cuadro de diálogo Cancelar
  CancelSend = False
  frmCancelSend.Label1.Caption =
    "Transmitiendo archivo de texto - " + Temp
  frmCancelSend.Show

    ' Lee el archivo en bloques del tamaño del buffer de transmisión
  BSize = MSComm1.OutBufferSize
  LF& = LOF(hSend)
  Do Until EOF(hSend) Or CancelSend
    If LF& - Loc(hSend) <= BSize Then
      BSize = LF& - Loc(hSend) + 1
    End If

    ' Lee un bloque de datos
    Temp = Space$(BSize)
    Get hSend, , Temp

    ' Transmite el bloque
    MSComm1.Output = Temp
    If Err Then
      MsgBox Error$, 48
      Exit Do
    End If

    ' Espera a que se envíen todos los datos
    Do
      Ret = DoEvents()
    Loop Until MSComm1.OutBufferCount = 0
     Or CancelSend
  Loop
End If

Close hSend
mnuSendText.Enabled = True
tbrToolBar.Buttons("TransmitTextFile").Enabled
    = True
CancelSend = True
frmCancelSend.Hide
End Sub
```

Este procedimiento que estamos tratando en el artículo establece la propiedad *SThreshold*, que determina el número máximo de caracteres que deben estar esperando en el buffer de salida para que se establezca **comEvSend** en la propiedad *CommEvent* y se dispare el evento *OnComm*.

```
Private Sub mnuSThreshold_Click()
  On Error Resume Next

  Temp = InputBox$("Introduzca el nuevo valor
    de SThreshold", "SThreshold",
    Str$(MSComm1.SThreshold))
  If Len(Temp) Then
    MSComm1.SThreshold = Val(Temp)
    If Err Then MsgBox Error$, 48
  End If
End Sub
```

El siguiente procedimiento agrega datos a la propiedad *Text* del control **Term**. También filtra los caracteres de control, como RETROCESO, retorno de carro y avances de línea, y escribe datos en un archivo de registro. Los caracteres RETROCESO eliminan el carácter situado a su izquierda, ya sea en la propiedad *Text* o en la cadena pasada. Se agregan caracteres de avance de línea a todos los retornos de carro. El tamaño de la propiedad *Text* del control **Term** también se controla para que nunca exceda de **MAXTERMSIZE** caracteres.

```
Private Static Sub ShowData(Term As Control,
    Data As String)
  On Error GoTo Handler
  Const MAXTERMSIZE = 16000
  Dim TermSize As Long, i

    ' Se asegura que el texto existente no se haga demasiado largo
  TermSize = Len(Term.Text)
  If TermSize > MAXTERMSIZE Then
    Term.Text = Mid$(Term.Text, 4097)
    TermSize = Len(Term.Text)
  End If

    ' Apunta al final de los datos de Term
  Term.SelStart = TermSize

    ' Filtra y procesa los caracteres RETROCESO
  Do
    i = InStr(Data, Chr$(8))
    If i Then
      If i = 1 Then
        Term.SelStart = TermSize - 1
        Term.SelLength = 1
        Data = Mid$(Data, i + 1)
      Else
       Data = Left$(Data, i - 2) & Mid$(Data, i + 1)
      End If
    End If
  Loop While i

    ' Elimina los avances de línea
  Do
    i = InStr(Data, Chr$(10))
    If i Then
      Data = Left$(Data, i - 1) & Mid$(Data, i +
      1)
    End If
  Loop While i

    ' Se asegura de que todos los retornos de carro tengan un avance de línea
  i = 1
  Do
    i = InStr(i, Data, Chr$(13))
    If i Then
      Data = Left$(Data, i) & Chr$(10) &
      Mid$(Data, i + 1)
      i = i + 1
    End If
  Loop While i

    ' Agrega los datos filtrados a la propiedad SelText
  Term.SelText = Data

    ' Registra los datos en un archivo si así se solicita
  If hLogFile Then
    i = 2
    Do
      Err = 0
      Put hLogFile, , Data
      If Err Then
        i = MsgBox(Error$, 21)
        If i = 2 Then
          mnuCloseLog_Click
        End If
      End If
    Loop While i <> 2
  End If
  Term.SelStart = Len(Term.Text)
Exit Sub

Handler:
  MsgBox Error$
  Resume Next
End Sub
```

Mediante este *Timer* chequeamos el estado de la comunicación actual.

```
Private Sub Timer2_Timer()
```

```
sbrStatus.Panels("Status").Text = "Estado:"
Timer2.Enabled = False
End Sub
```

Las pulsaciones interceptadas aquí se envían al control *MSComm*, donde se devuelven a través del evento *OnComm* (*comEvReceive*), y se muestran con el procedimiento *ShowData*.

```
Private Sub txtTerm_KeyPress(KeyAscii As
    Integer)
    ' Si el puerto está abierto...
  If MSComm1.PortOpen Then
    ' Envía la pulsación al puerto
    MSComm1.Output = Chr$(KeyAscii)

    ' Si ECHO no está activado, no hay
necesidad de que el control de texto
muestre la tecla. Normalmente, el
módem devolverá el carácter
    If Not Echo Then
    ' Sitúa la posición al final del termi-
nal
      txtTerm.SelStart = Len(txtTerm)
      KeyAscii = 0
    End If
  End If

End Sub
```

Gestionamos todos los eventos posibles del menú desde este procedimiento, llamando al que corresponda:

```
Private Sub tbrToolBar_ButtonClick(ByVal Button
    As ComctlLib.Button)
Select Case Button.Key
Case "OpenLogFile"
  Call mnuOpenLog_Click
Case "CloseLogFile"
  Call mnuCloseLog_Click
Case "DialPhoneNumber"
  Call mnuDial_Click
Case "HangUpPhone"
  Call mnuHangup_Click
Case "Properties"
  Call mnuProperties_Click
Case "TransmitTextFile"
  Call mnuSendText_Click
End Select
End Sub
```

Este *Timer* muestra la hora de conexión:

```
Private Sub Timer1_Timer()
 sbrStatus.Panels("ConnectTime").Text =
    Format(Now - StartTime, "hh:nn:ss") & " "
End Sub
```

```
' Llama a esta función para iniciar el
cronómetro ConnectTime
Private Sub StartTiming()
  StartTime = Now
  Timer1.Enabled = True
End Sub
' Llama a esta función para detener
el cronometraje
Private Sub StopTiming()
  Timer1.Enabled = False
  sbrStatus.Panels("ConnectTime").Text = ""
End Sub
```

CONCLUSIÓN FINAL

Como habréis podido apreciar, el control MSCOMM puede tener muchas posibilidades de programación, tan sólo hay que poner un poco de ingenio por nuestra parte para sacarle todo el provecho posible a la aplicación.

Como muestra de otro tipo de programas que se pueden crear con este completo control, podemos hacer que una aplicación responda a las llamadas entrantes de forma automática mediante la reproducción de un archivo WAV, al igual que si hubiera un contestador automático. Para ello, habría que detectar la llamada, descolgarla automáticamente con el módem y reproducir el sonido. Seguidamente, enviaríamos un comando al módem para que cortara la conexión.

Algo más complejo sería grabar los mensajes del que llama, aunque desde luego todo es posible.

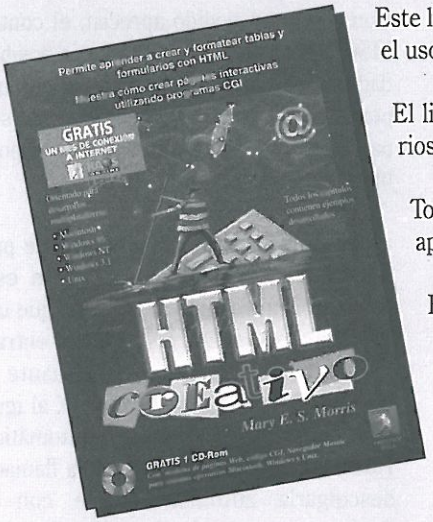
Porque su futuro ya está cambiando. Porque están aprovechando sus propios recursos. Porque, trabajando juntos, estamos contribuyendo para que dejen de ser el Tercer Mundo.

Desde 1981
Ayuda en Acción
Sembrando Futuro

LIBROS

HTML Creativo

El lenguaje marcado de hipertexto (HTML) es el de World Wide Web, la parte de un fenómeno de rápida evolución, llamado Internet, que crece más aprisa.



Este libro trata de todos los aspectos básicos de construcción de HTML, poniendo énfasis en el uso de los guiones CGI y formularios para crear páginas web de manera personalizada.

El libro HTML Creativo permite al usuario aprender a crear, formatear tablas y formularios con HTML, y muestra cómo crear páginas interactivas utilizando programas CGI.

Todos los capítulos del libro contienen amplios ejemplos desarrollados para facilitar el aprendizaje del lector.

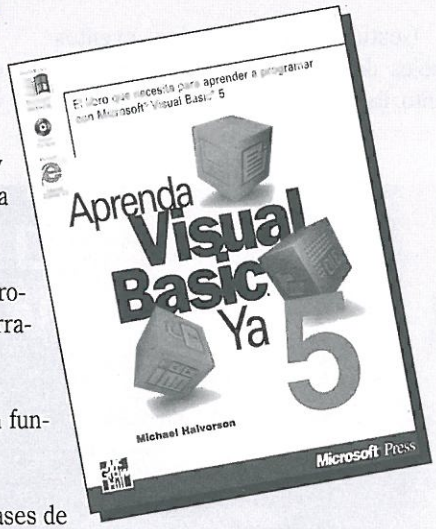
Este manual está acompañado de un CD-ROM con el código HTML de los ejemplos del libro, los programas CGI utilizados en los ejemplos, Microsoft Windows Server y el navegador Mosaic para sistemas Windows, UNIX y Macintosh.

Editorial: Prentice Hall
Nº páginas: 248
Nivel: Intermedio-avanzado

Autor: Mary E.S. Morris
Idioma: Español
Precio: 3.900 Ptas.(I.V.A. inc.)

Aprenda Visual Basic 5 Ya

Este libro está pensado y diseñado teniendo siempre en mente al nuevo usuario de Visual Basic, por lo que no se necesita experiencia para poder utilizarlo.



Una serie de ejercicios fáciles y sencillos le ayudarán a diseñar con rapidez y facilidad una interfaz de usuario y desarrollar el código de los programas para Microsoft Windows 95.

Además Aprenda Visual Basic 5 Ya le enseñará a trabajar con el entorno de programación de Visual Basic, podrá utilizar los controles contenidos en la caja de herramientas, las propiedades de los objetos y el código del programa.

También aprenderá a crear una interfaz gráfica de usuario y personalizada con funciones de animación de "arrastrar y soltar".

Le permitirá averiguar cómo se trabaja con mapas de bits, archivos de texto, bases de datos y objetos de aplicaciones Microsoft Office.

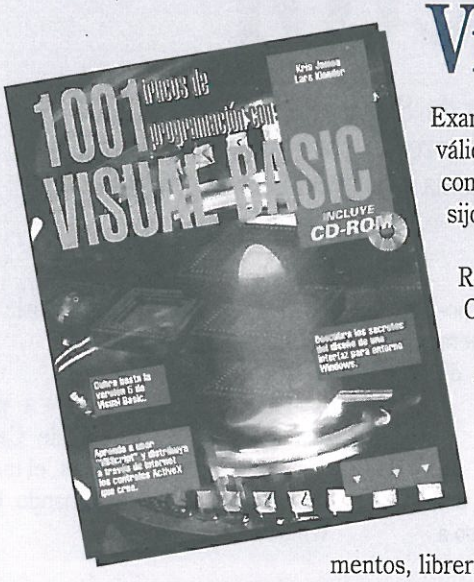
El libro incluye un CD-ROM que contiene los archivos de las prácticas y todos los ejemplos del libro.

Editorial: Mc Graw Hill
Nº páginas: 408
Nivel: Principiante

Autor: Michael Halvorson
Idioma: Español
Precio: 5.900 Ptas.(I.V.A. inc.)

LIBROS

1001 trucos de programación con Visual Basic



Examine cada una de las posibilidades que le ofrece Visual Basic 5 con este libro válido tanto para el programador que se enfrenta por primera vez a este lenguaje como para el experto en Visual Basic que quiere saber algo más sobre los entresijos de este potente y lucrativo lenguaje de programación.

Realice la gestión de sus bases de datos a través de objetos DAO y del estándar ODBC que permite gestionar también las bases de datos remotas.

Aprenderá a crear informes altamente profesionales y a asignarles el formato deseado directamente desde sus programas en Visual Basic. Podrá acelerar el diseño de sus programas aprovechando las nuevas características que le ofrece el nuevo Entorno Interactivo de Desarrollo de Visual Basic 5.0.

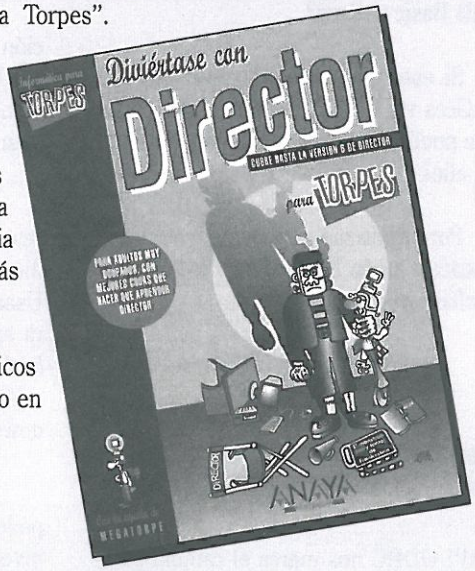
Aprenda a gestionar de forma óptima sus propios controles ActiveX, documentos, librerías y servidores; conozca, gracias a este manual, todos los secretos para controlar las comunicaciones con Visual Basic.

Editorial: Anaya Multimedia
Nº páginas: 1008
Nivel: Intermedio-avanzado

Autor: Kris Jamsa y Lars Klander
Idioma: Español
Precio: 6.495 Ptas.(I.V.A. inc.)

Diviértase con Director para torpes

Director para Torpes es otro nuevo libro de la colección "Informática para Torpes". La colección está dirigida a todas aquellas personas que encuentran difícil aprender informática, ya sea por falta de tiempo, por su temor ante los ordenadores o por la obligación de manejar un PC.



Con este libro el lector obtendrá de una forma amena y divertida, todos los conocimientos necesarios para crear sus propias aplicaciones con Director. Para desdramatizar el contenido, eliminando así cualquier tipo de animadversión hacia la informática, se han añadido toques del mejor humor, que le harán mucho más sencilla la tarea de aprender informática.

Además los libros están ilustrados por uno de los mejores humoristas gráficos de nuestro país: Forges, que ha creado a Cosme Romerales protagonista del libro en cuestión.

Editorial: Anaya Multimedia
Nº páginas: 288
Nivel: Principiante-Intermedio

Autor: José María Delgado Cabrera
Idioma: Español
Precio: 2.495 Ptas.(I.V.A. inc.)

Correo del lector

En esta sección, los lectores de SÓLO PROGRAMADORES tienen la oportunidad de hallar respuesta a los problemas que puedan tener en cualquier tema relacionado con la programación.

Pregunta

Queridos amigos de Sólo Programadores, me dirijo a vosotros por dos motivos, en primer lugar para felicitaros por vuestra publicación, la cual, en mí opinión, está a años luz del resto de las que abordan el tema de la programación.

En segundo lugar, y en relación con el artículo que publicabais en el número 46 acerca de PostGres, la base de datos de Linux, por cierto muy ameno y práctico, me gustaría que me contestarais a las siguiente pregunta: ¿es esta base de datos adecuada para desarrollar aplicaciones cliente-servidor que ataquen a dicha base de datos bajo clientes basados en Delphi, Visula Basic u otros?

Si esto es así, ¿es posible atacar a PostGres vía ODBC, y en este caso, dónde se puede conseguir el driver adecuado para ello?

Por último me gustaría que me recomendárais algún libro sobre PostGres en castellano si es que existe.

José Manuel Izquierdo Martín.
Ávila

Respuesta

El API ODBC nos marca el camino a seguir para conectarnos a una fuente de datos ODBC. Esta forma de conectarnos podría ser cualquier cosa, desde un fichero de texto, hasta un complejo sistema de gestión de bases de datos Oracle.

Este acceso llegará procedente de drivers ODBC o de drivers específicos del fabricante que proporcionen este acceso a datos. PostODBC es este tipo de driver, tanto como otros también disponibles como los drivers OpenLink.

El driver PostODBC está disponible con la distribución de PostGreSQL que se incluía en el CD-ROM del nº46 en el directorio **src/interfaces/odbc**, el otro, también popular está disponible en: http://www.openlinksw.com/

Una vez que has escrito una aplicación ODBC, deberías de poder conectarte a CUALQUIER base de datos, prescindiendo del vendedor, el esquema es el mismo.

Por ejemplo: podrías tener servidores *MS SQL Server* y *PostGreSQL* conteniendo exactamente los mismos datos. Usando ODBC, tu aplicación Windows hará exactamente las mismas llamadas y el *back end* de la fuente de datos tendrá el mismo aspecto (para la aplicación Windows).

En el mundo real, las diferencias que podemos establecer entre los drivers y el nivel de compatibilidad ODBC disminuye su potencial:

- Access, Delphi, y Visual Basic soportan ODBC directamente.
- Bajo C++ como Visual C++ puedes usar el API ODBC de C++, usando la clase CrecordSet, el mejor camino si estás programando bajo Windows NT.

Por otra parte no tenemos noticia de ningún libro sobre este tema en castellano, pero existen varias listas de correo (mailing lists) disponibles para formular preguntas en castellano sobre éste y otros temas que se te presenten, sus direcciones son las siguientes: pgsql-ayuda@campus.iztacala.unam.mx para enviar las preguntas y para darse de alta y baja en el servicio enviar un mensaje a majordomo@campus.iztacala.unam.mx en el que figure (en el cuerpo del mensaje) y en una sola línea "inscripción pgsql-ayuda" o "cancelación pgsql-ayuda".

Pregunta

Ante todo deciros, aunque sea repetitivo, que tenéis la mejor revista sobre programación en castellano que conozco. Soy subscriptor desde hace tiempo, y realmente me encanta la revista.

Soy estudiante de informática y hace tiempo, explorando las opciones de uno de los compiladores de Borland vi una opción para configurar un segundo monitor para tener en él la ventana de depurado.

Noticias frescas, cada día

TOWER
COMMUNICATIONS

EN INFORMÁTICA MARCAMOS EL CAMINO

NET MAGAZINE
PC MEDIA
SÓLO PROGRAMADORES
PC PLAYER

Estuve preguntando e investigando y al final me dijeron que colocando una tarjeta como monocroma y otra como color funcionaba. Pero no conseguí configurarlo correctamente.

Ahora tengo un P75 con una tarjeta de vídeo S3 trio 64 con 2Mb y tengo además un par de tarjetas VGA y SVGA Cirrus Logic, así como un monitor IBM blanco y negro (VGA) y otro de color (SVGA). ¿Puedo hacer algo con esto?. Si no puedo ¿Qué necesito para hacerlo?.

Gracias por adelantado.

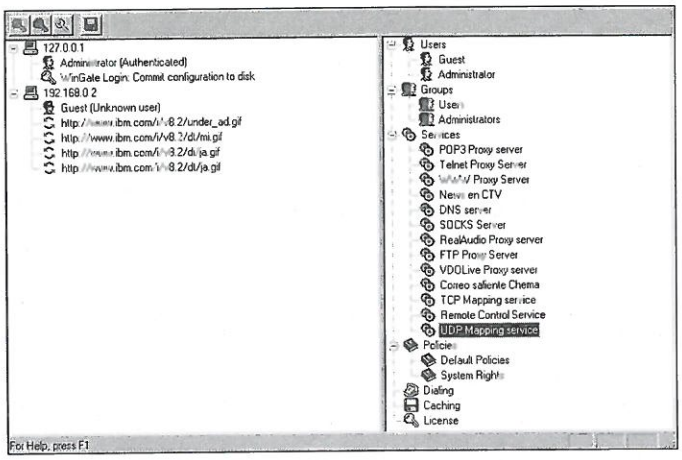
Fernando París.

Respuesta

Tu pregunta resulta bastante interesante, actualmente, no importa lo grande que sea nuestra pantalla, parece que nunca es suficiente. Hay dos opciones, o ahorrar y comprar un monitor de 35" o un segundo monitor.

La opción que nos propones es posible, pero difícil de llevar a cabo por las restricciones en hardware que Windows 95 hace al respecto. Para empezar no te valdrían las tarjetas que pretendes usar como secundarias, ya que es recomendable que sea una tarjeta monocroma nativa, como la famosa y caduca Hércules (difícil de conseguir), por otra parte es posible que ni aún así te funcionase, ya que los drivers de vídeo suelen se bastante quisquillosos a la hora de compartir su información y una vez conseguido eso, puede que el software que utilizases no fuese compatible con la combinación anterior.

Con un poco de paciencia tu problema se verá resuelto en breve, con la llegada de Windows 98 y Windows NT 5.0 que prometen incluir un conjunto de características que permitirán múltiples dispositivos de visualización; esto es, múltiples tarjetas de vídeo y monitores en la misma máquina, todas formando parte del mismo escritorio virtual y con soporte transparente para el usuario dentro del sistema operativo.



Wingate gestiona el acceso de un usuario a la sede de IBM.

El lanzamiento del nuevo sistema operativo de Microsoft será en todo el mundo el 25 de Junio, pero la opción de múltiples monitores y visualizaciones ya fue adelantada en su versión Beta, Menphis, con excelentes resultados.

Pregunta

Me dirijo a ustedes con el motivo de solicitarles información con respecto al software(FREE) necesario para la implementación de una red interna con salida a Internet.

La configuración de la máquina es la siguiente: 2 computadoras 586 con 8 MB RAM y placa de RED; 1 computadora 586 con 32 Mb RAM con módem externo US ROBOTIC SPORTER 33.6 Kb con placa de red.

La idea es utilizar la máquina con módem como compuerta de salida hacia Internet de las otras dos y utilizar WINDOWS 95. ¿Podrían indicarme además, si el sistema operativo Linux puede realizar este trabajo como compuerta de salida hacia Internet con todos los servicios que este brinda?.

Muchas gracias anticipadas, no me gustaría despedirme sin decirles que la revista Sólo Programadores, de la que soy asiduo desde hace tiempo, es muy buena y práctica con lo cual se convierte en una excelente herramienta de aprendizaje. Sigan así.

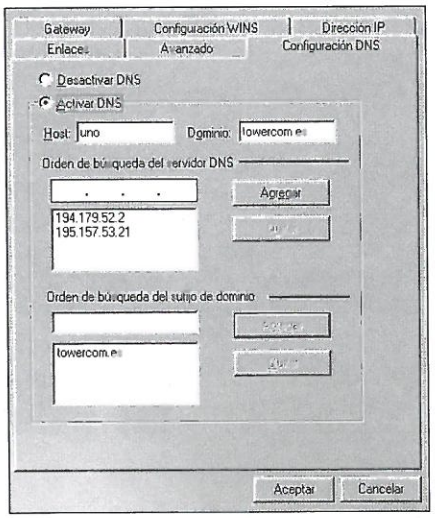
Germán Walter Toconá

Respuesta

La respuesta a tu pregunta es realmente fácil y si eres lector habitual de la revista poseerás la herramienta adecuada para incorporar su futura Intranet en Internet.

La configuración que nos propones exige que la red local a la que quiere dar salida a Internet funcione sobre el protocolo TCP/IP, éste está disponible en el CD-ROM de instalación de Windows 95 que es el sistema operativo que nos indicas (configuración->red->añadir protocolo->Microsoft->TCP/IP).

Ésto no quiere decir que al mismo tiempo que se usa este protocolo no puedas estar usando otro a tu gusto, por



Configuración TPC/IP de la pasarela a la red Internet.

ca diferencia con el producto original es que la licencia expira el primero de Septiembre del 98. NDS for NT posibilita la inter-operatividad entre las redes IntranetWare y Windows NT.

NDS for NT relocaliza los dominios NT dentro del árbol NDS. Dicha relocalización, transparente para el usuario, permite a los dominios NT ser objetos NDS manejables, llamados objetos de Dominio. De esta forma, los administradores pueden gestionar dominios usando tanto Novell como Microsoft.

Además incorpora el Mailbox Manager for Exchange, que permite administrar Microsoft Exchange con la herramienta de Novell NWAdmin, permitiendo la creación, edición y borrado de buzones de correo de Exchange sin tener acceso al resto de las capacidades del administrador de Exchange.

Para instalarlo hay que copiar los auto-extraibles que contiene el directorio \REDES\NDS en el servidor NT que debe ser el controlador primario del dominio y tener instalada la IntranetWare.

BorderManager FastCache

Utilidad de cacheado de Internet. Soporta NT, Unix, Macintosh y NetWare. BorderManager acelera dramáticamente el acceso de los usuarios a la información más popular de la Web, reduciendo los requerimientos de ancho de banda y aumentando el rendimiento de los servidores Web que poseamos.

Ésta es una versión de prueba antes de comprar, que caduca a los 45 días después de su lanzamiento. Están incluidas instrucciones de cómo instalar la licencia una vez que compremos el producto sin la necesidad de desinstalar la versión de prueba existente.

Requerimientos:

- NetWare 4.11 o IntranetWare instalado en el servidor.
- Support Pack 4.0 de Novell (la cual proporcionamos)
- TCP/IP disponible en el servidor
- Pasarela Novell IPX/IP desactivada en la estación donde vayamos a instalar BorderManager
- Clientes IntranetWare
- NetWare Administrator ejecutado al menos una vez en el cliente donde lo vayamos a instalar.
- 45MB de espacio en el disco del servidor.
- 32MB de RAM.

Radius Services for NDS

Radius Services for NDS permite la administración de los usuarios entrantes en Novell Directory Services para redes cuyos servidores cumplan con la normativa Radius incluyendo Ascend, Cisco, Microsoft, Shiva, Telebit, US Robotics y otros muchos.

IntranetWare HostPublisher

Es el primer producto específicamente diseñado para permitir a los usuarios acceso completo, en todos los sentidos, a las aplicaciones de un mainframe IBM, desde cualquier navegador basado en TCP/IP, independientemente del sistema operativo sobre el que se ejecute. HostPublisher proporciona automáticamente una interfaz gráfica interpretando el flujo de datos del mainframe 3270, traduciéndolo, en tiempo real, a HTML.

Para instalarlo tendremos que descomprimir el fichero proporcionado en el directorio \REDES\IWHP\ en el servidor.

LDAP Services for NDS v1.0

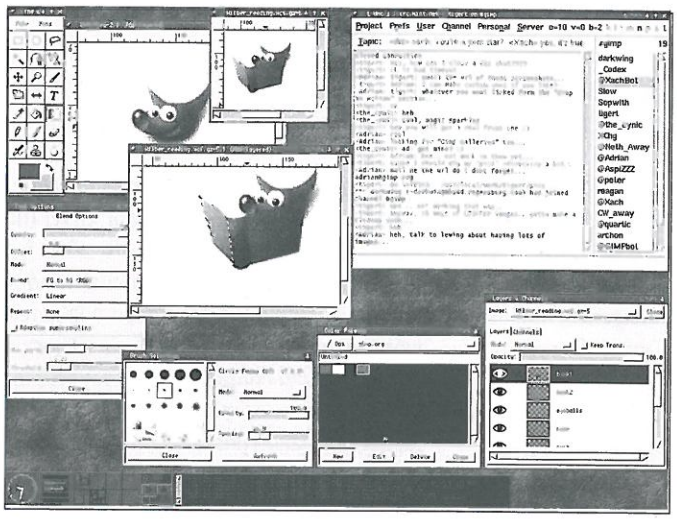
LDAP Services for NDS publica la información almacenada en el NDS (Novell Directory Services) para clientes basados en el protocolo de acceso a directorios simplificado (LDAP). LDAP (Lightweight Directory Access Protocol es un protocolo en desarrollo para acceder a información de directorios como la de usuario y de Host.

Se espera que este protocolo se convierta en una característica estándar de los navegadores de Internet.

UNIX

GNU Network Object Model Environment

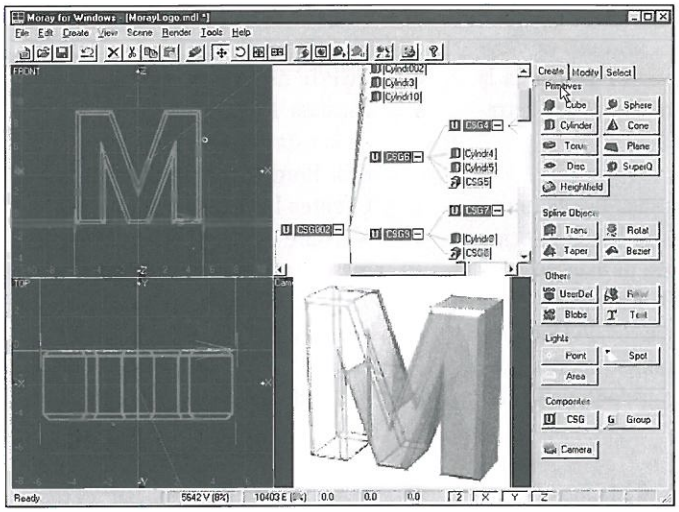
El proyecto que presenta GNOME tiene como principal objetivo construir un escritorio completo, amable con el usuario y basado enteramente en programas de libre distribución.



GIMP nos ayuda en el retoque fotográfico y en la creación de imágenes y su composición.

Este proyecto forma parte de otro, GNU y también forma parte del movimiento de Internet OpenSource. El escritorio consta de pequeñas utilidades y grandes aplicaciones que poseen un consistente aspecto y forma de utilización, lo que los "sajones" han dado en llamar 'look and feel'. GNOME usa GTK+ como herramienta para la creación del interfaz con el usuario de todas las aplicaciones que cumplan con el estándar GNOME.

Ésta es la intención inicial del proyecto, que está sujeto a discusión y modificación, por supuesto.



Moray v3.0 permite una representación en dos vistas: 2D y 3D.

NetShow Theater Server posee una arquitectura tolerante a fallos y distribuida

Todo lo referente al estado del mismo, así como todas las aplicaciones que hay en desarrollo y ya desarrolladas que cumplen con GNOME, se encuentran disponibles en binarios instalables para varias distribuciones como RedHat, así como en forma de fuentes, en el directorio \UNIX\GNOME.

GNU Image Manipulation Program

Es una pieza de software de libre distribución adecuada para tareas como el retoque fotográfico, creación de imágenes y su composición.

Es muy versátil, ya que puede ser usada como un simple programa, como una aplicación profesional de retoque fotográfico o como un sistema en línea de proceso por lotes de imágenes; incluso como un productor en masa de renders, conversor de formatos gráficos, etc.

GIMP es extremadamente expandible y extensible. Está diseñado para ser aumentado con plug-ins y extensiones para hacer casi todo. La interfaz avanzada con características de script permite todo, desde una simple tarea, al más complejo procedimiento de manipulación de imágenes puede ser fácilmente guiado por el script.

Para más información ver el directorio \UNIX\GIMP.

GTK+

GTK quiere decir GIMP Toolkit, es una librería para la creación de entornos gráficos para el usuario en el sistema X Window. Está diseñado para ser pequeño, eficiente y flexible. Está escrito en C con una aproximación a la orientación a objetos. Existen ports para C++, Perl, Python, Objective C y Guile.Ha sido lanzado bajo licencia GNU por lo que es de libre distribución.

Proyecto Egcs

EGCS es un paso experimental en el desarrollo de GCC, el compilador GNU (de libre distribución). El proyecto aúna los esfuerzos por hacer fundamental la idea de que la visibilidad del código y la sumisión a la red son vitales para la mejora a largo plazo del compilador. El proyecto integra variaciones y parches a GCC que andan circulando por la red, incluyendo algunos front-ends como g77 (Fortran). Viene trabajando en colaboración con los sistemas operativos gratuitos (Linux, BSD) en los que GCC es una pieza fundamental en la infrestructura.

VARIOS

NetShow Theater Server

Microsoft NetShow Theater Server reparte multimedia de alta calidad a cientos de clientes en cualquier momento y allí donde la red alcance.

- Tiene la potencia y flexibilidad para cumplir con la demanda de aplicaciones interactivas en redes públicas y privadas de gran ancho de banda.
- Plataforma multimedia ideal para el desarrollo de soluciones a medida cuando se necesita un rico contenido en audio y vídeo.
- Es escalable desde unos pocos flujos de datos a miles de ellos.
- Su arquitectura sobre PC tolerante a fallos y distribuida, lo hace, en cuanto a precio, muy rentable para aplicaciones de misión crítica.
- La gestión del contenido y las herramientas de administración lo hacen fácil de manejar, mantener y acomodar a nuestras necesidades.

Requerimientos:

- Ordenador Intel Pentium/166 con 32MB y tarjeta de vídeo adecuada.

BUSCAMOS PROFESIONALES

Si te gusta escribir y tienes experiencia en algunos de estos campos de la informática

HARDWARE:

Tarjetas gráficas, modems, PC Cards, cámaras digitales, escáneres, tarjetas de sonido, impresoras, monitores, pantallas planas, dispositivos de almacenamiento, vídeo digital...

SOFTWARE:

Ofimática en general, aplicaciones profesionales de diseño, Internet, autoedición, multimedia, animaciones 3D, programación, sonido, edición de vídeo, antivirus, edición Web, juegos...

INTERNET:

Navegación, videoconferencia, conectividad, redes, ActiveX, *plug-ins*, Shockwave, RDSI, sistemas abiertos, comercio electrónico, sistemas de seguridad, intranet, extranet...

Envíanos tu C.V. a la siguiente dirección

Tower Communications
Apartado de correos 54.283
28080 - Madrid

(Ref.: Hard - Soft - Int)

- Red Ethernet o ATM.
- Microsoft Windows NT WorkStation v4.0 y Service Pack 3 o Windows95 con el último Service Pack.

POV-Ray v3.0

Es una herramienta de alta calidad totalmente gratuita, para la creación de gráficos tridimensionales. Tiene licencia GNU lo cual le convierte en el programa de rendering 3D más utilizado por los usuarios de Internet. Es potente y flexible, permitiendo al programador realizar sus propios scripts para automatizar sus operaciones. El rendimiento es óptimo, sacándole el máximo partido a la máquina.

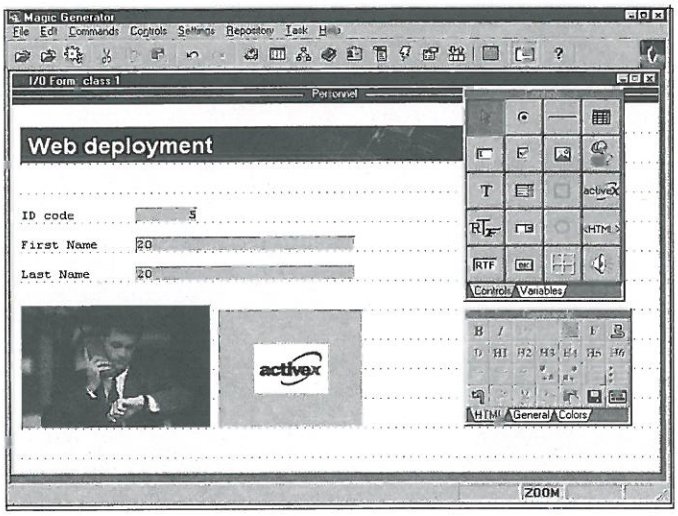
Moray es un modelador de mallas interactivo que crea escenarios POV-Ray

En el directorio \VARIOS\POV-RAY30 encontrará versiones para casi todos los sistemas operativos, documentación abundante y ejemplos.

Polyray v1.8

Polyray es una combinación de un extenso conjunto de primitivas de modelado y un render de fácil comprensión. Los modelos son especificados en un fichero ASCII, analizados y dibujados. Un IBM-AT o compatible es la única plataforma soportada. Los requerimientos mínimos son un 80386/80387 con 2MB.

Polyray proporciona opciones muy interesantes, la más destacable es la libertad en el uso de expresiones en el fichero de modelado y la capacidad de dibujar superficies matemáticas muy complejas.



Con Magic podemos crear aplicaciones de misión crítica mediante un entorno RADD.

Moray v3.0

Moray es un modelador de mallas (wireframes) interactivo que permite crear escenarios POV-Ray (o Polyray), permitiendo ver el aspecto de la escena mientras lo construyes. No hay necesidad de editar los scripts de POV-Ray, a no ser que realmente se quiera. Moray muestra una representación en malla de la escena en dos vistas 2D y vistas de cámara 3D.

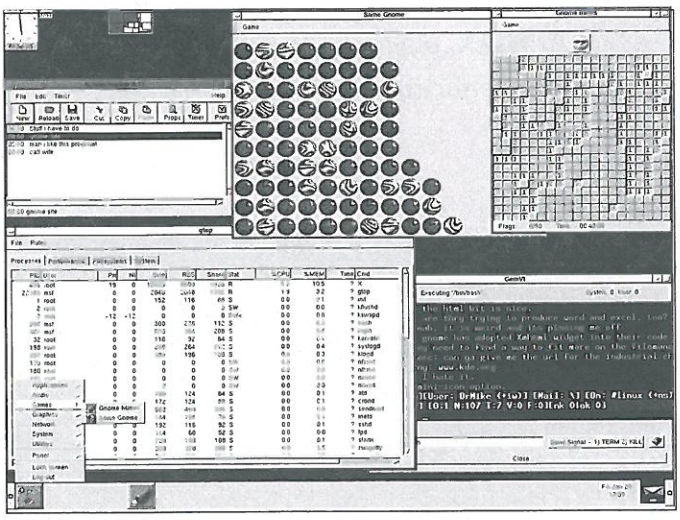
Moray v3.0 es la última versión de este modelador para usar conjuntamente con POV-Ray y Polyray. Está mejorado para trabajar con los 32 bits de Windows. Sólo funciona en Windows 95/98 y NT 4.0.

En el directorio \VARIOS\MORAY\MORAY25 existe una versión anterior, la 2.5 para DOS. También facilitamos una librería con obras de usuarios expertos de Moray, a fín de que sirvan de ayuda en el directorio \VARIOS\MORAY\LIB.

Xara3D

Los programas tradicionales de 3D son notablemente difíciles de usar y comprender, al contrario que mucho software moderno, que ofrece un montón de utilidades que nunca van a ser usadas y que llevan mucho tiempo de comprensión y aprendizaje. Xara3D ha sido creada para llevar a cabo una única tarea, y la realiza con estilo, velocidad y simplicidad.

Xara3D fue diseñada para construir encabezamientos 3D de alta calidad para las páginas Web. Todas las imágenes están suavizadas con calidad profesional. Xara3D puede además realizar, en segundos, sencillos GIFs animados de gran estilo, o incluso secuencias AVI.



GNOME usa GTK+ como herramienta para la creación del interfaz.